Уральский федеральный университет
имени первого Президента России Б.Н.Ельцина
Институт радиоэлектроники и информационных технологий — РТФ

И. А. СПИЦИНА
К. А. АКСЕНОВ

МУЛЬТИАГЕНТНЫЙ МЕТОД АНАЛИЗА И СИНТЕЗА ИНФОРМАЦИОННЫХ СИСТЕМ

Учебное пособие

Министерство образования и науки Российской Федерации
Уральский федеральный университет
имени первого Президента России Б. Н. Ельцина

И. А. Спицина, К. А. Аксенов

Мультиагентный метод анализа и синтеза информационных систем

Учебное пособие

Рекомендовано методическим советом
Уральского федерального университета
для студентов вуза, обучающихся
по направлению подготовки
09.04.01 — Информатика и вычислительная техника

Екатеринбург
Издательство Уральского университета
2017

УДК 004.8(075.8)
ББК 32.813я73
С71

Рецензенты:
кафедра «Высшая и прикладная математика» Уральского государственного университета путей сообщения (завкафедрой проф., д-р физ.-мат. наук *Г. А. Тимофеева*);
доц., канд. техн. наук *В. Ф. Ярчук* (начальник программно-технологического отдела ООО «ТЭКСИ-Консалтинг»).
Научный редактор — проф., д-р техн. наук *Л. Г. Доросинский*

Спицина, И. А.
С71 Мультиагентный метод анализа и синтеза информационных систем : учебное пособие / И. А. Спицина, К. А. Аксенов. — Екатеринбург : Изд-во Урал. ун-та, 2017. — 92 с.

ISBN 978-5-7996-2038-7

Отражены аспекты принятия решений при разработке информационных систем. Основное внимание уделено построению модели информационной системы с использованием автоматизированных средств поддержки принятия решений. Рассмотрены методы разработки организационно-технических систем и представлен анализ существующих CASE-средств. Предназначено для студентов дневной и дистанционной форм обучения направления 09.04.01 — Информатика и вычислительная техника (магистр), модуль «Применение аналитики и графики в информационных системах (ИС)».

Библиогр.: 23 назв. Табл. 4. Рис. 46.

УДК 004.8(075.8)
ББК 32.813я73

Учебное издание

Спицина Ирина Александровна, **Аксенов** Константин Александрович

МУЛЬТИАГЕНТНЫЙ МЕТОД АНАЛИЗА И СИНТЕЗА ИНФОРМАЦИОННЫХ СИСТЕМ

Редактор *И. В. Коршунова*
Верстка *О. П. Игнатьевой*

Подписано в печать 17.04.2017. Формат 70×100/16. Бумага писчая. Печать цифровая. Гарнитура Newton.
Уч.-изд. л. 5,0. Усл. печ. л. 7,4. Тираж 50 экз. Заказ 70

Издательство Уральского университета
Редакционно-издательский отдел ИПЦ УрФУ
620049, Екатеринбург, ул. С. Ковалевской, 5. Тел.: 8 (343)375-48-25, 375-46-85, 374-19-41. E-mail: rio@urfu.ru

Отпечатано в Издательско-полиграфическом центре УрФУ
620075, Екатеринбург, ул. Тургенева, 4. Тел.: 8 (343) 350-56-64, 350-90-13. Факс: 8 (343) 358-93-06
E-mail: press-urfu@mail.ru

ISBN 978-5-7996-2038-7

Оглавление

Список основных сокращений

BPMN	(Business Process Model and Notation) — нотация и модель бизнес-процессов
CASE	(Computer Aided Software Engineering) — автоматизированная разработка ПО
UML	(Unified Modeling Language) — унифицированный язык моделирования
БД	– база данных
БЗ	– база знаний
БП	– бизнес-процесс
ИА	– интеллектуальные агенты
ИИ	– искусственный интеллект
ИМ	– имитационное моделирование
ИС	– информационная система
ИТ	– информационные технологии
КГ	– концептуальный граф
КМПО	– концептуальная модель предметной области
ЛПР	– лицо, принимающее решение
МАС	– мультиагентные системы
МЛВ	– машина логического вывода
МППР	– мультиагентные процессы преобразования ресурсов
ООП	– объектно-ориентированный подход
ОТС	– организационно-технические системы
ПИ	– пользовательский интерфейс
ПО	– программное обеспечение
ППР	– поддержка принятия решения
РБП	– реинжиниринг бизнес-процессов
СДМС	– система динамического моделирования ситуаций
СМО	– системы массового обслуживания
СППР	– система поддержки принятия решений
СУБД	– система управления базами данных
ТЗ	– техническое задание
ФК	– фрейм-концепт
ЭС	– экспертная система

Предисловие

Учебное пособие посвящено методу поддержки принятия решений в области создания информационных систем на основе мультиагентного подхода. Объектом автоматизации выступает организационно-техническая система, которая представляет собой совокупность организационной структуры и находящихся в ее распоряжении технических средств, т. е. совместно рассматривается человек и информационная система. Разработка ИС является мероприятием с высокой степенью риска. Согласно исследованиям, только 22 % проектов, длящихся более двух лет, завершаются в установленный срок. Одна из причин такого явления — это искажение и потеря информации о разрабатываемой ИС и особенностях процесса в цепочке ее передачи пользователь — аналитик — разработчик. Техническое задание должно являться связующим звеном между ними. Однако оно недостаточно полно отражает предметную область ОТС (в части процессов согласования и принятия решений), а также не решает вопрос увязывания ожиданий пользователя с требованиями к ИС. Из всего ТЗ пользователю понятен раздел, описывающий функции системы, с ИС пользователь отождествляет визуальный пользовательский интерфейс.

Успешность разработки ИС во многом определяется проработанностью методологического подхода, используемого в процессе проектирования. При этом следует отметить следующие моменты. Во-первых, существующие методики и инструментальные средства не дают единой модели информационной системы как с точки зрения разработчика, так и пользователя — предметного специалиста. Во-вторых, для ОТС характерны процессы принятия решений, которые предполагают работу со знаниями, формализуются сценариями и в ряде случаев предполагают согласование решений. Существующие методики не позволяют в комплексе решать вопросы формализации и информатизации процессов принятия решений. В-третьих, для анализа, совершенство-

вания и реинжиниринга бизнес-процессов в ОТС используются средства имитационного и мультиагентного моделирования. Однако применение данных средств на этапах автоматизации и информатизации до сих пор ограничено в силу двух причин: во-первых, затраты на разработку имитационной модели; во-вторых, отсутствие возможностей использовать полученные результаты и знания на этапах автоматизации. Эффект от информатизации будет намного выше, если решать задачу автоматизации совместно с задачей совершенствования БП.

Большой вклад в рассматриваемую тему внесли работы следующих исследователей: К. А. Аксенова, Д. В. Александрова, Б. Боэма, Г. Буча, А. М. Вендрова, К. Гейна, С. Л. Гольдштейна, В. И. Городецкого, Г. Н. Калянова, О. В. Карсаева, Б. И. Клебанова, М. Минского, Е. Г. Ойхмана, Э. В. Попова, Дж. Рамбо, У. Ройса, Т. Сарсона, П. О. Скобелева, А. Ю. Филипповича, М. Хаммера, Дж. Чампи, А. Н. Швецова, N. R. Jennings, M. J. Wooldridge.

В настоящее время существуют различные подходы к разработке. Структурный подход (IDEF0, DFD) позволяет описать разрабатываемую систему в виде иерархии взаимосвязанных функций. Такое представление понятно аналитику и пользователю. Для анализа узких мест и динамических характеристик используется имитационное моделирование. При описании модели разрабатываемой системы, с точки зрения разработчика, применяют объектно-ориентированный подход (язык UML). Экспертные системы закрывают вопросы, связанные с описанием знаний и сценариев принятия решений. Использование мультиагентных систем позволяет автоматизировать процессы согласования решений и взаимодействие лиц, принимающих решения. Функции ЛПР выполняют программные агенты. Каждый из них в отдельности не закрывает всех вопросов, возникающих при автоматизации процессов ОТС. В связи с этим актуальным является анализ существующих динамических моделей процессов ОТС и моделей архитектуры информационных систем, а также создание на их основе метода ППР, совмещающего в себе эти подходы, и программного обеспечения для его реализации — системы поддержки принятия решений.

Идея учебного пособия заключается в интеграции методов и инструментальных средств ситуационного, мультиагентного, имитационного и экспертного моделирования с целью повысить эффективность принятия решений при ситуационном управлении преобразованием ресурсов.

Структура предлагаемого материала выглядит следующим образом.

Пособие состоит из трех глав, заключения и списка литературы.

В пособии обоснована необходимость автоматизации процесса принятия решений, приведен обзор методов моделирования мультиагентных процессов преобразования ресурсов, рассмотрены системы, близкие по функциональности к системам динамического моделирования ситуаций, и выполнен их сравнительный анализ, определены требования к СППР при разработке ИС. Излагаются принципы построения метода и СППР при разработке ИС, приведено описание данной системы, а также описаны принципы работы с ней.

Авторы благодарны Е. Ф. Смолий за оказанную неоценимую помощь при разработке и отладке СППР семейства BPsim. За предоставленную экспериментальную базу благодарим ООО «НПП “Системы автоматизации поддержки бизнеса”».

Авторы также благодарят всех аспирантов и студентов Уральского федерального университета имени первого Президента России Б. Н. Ельцина, принявших участие в отладке метода и сбора данных для эксперимента.

1. Процесс системного анализа при разработке информационных систем

1.1. Этапы системного анализа при разработке информационных систем

Основная проблема, возникающая при разработке информационной системы — это сложность понимания сразу всей системы в целом. Для решения этой проблемы целесообразно применять системный анализ. Такой подход позволяет получить целостное представление об объекте автоматизации и разрабатываемой ИС. Проведение системного анализа объекта автоматизации при разработке информационных систем можно разделить на следующие этапы:

- ✓ определение и назначение объекта автоматизации;
- ✓ определение целей разрабатываемой системы;
- ✓ анализ состояния внутренней и внешней среды автоматизируемого объекта и прогноз их изменений;
- ✓ построение и анализ моделей автоматизируемых процессов;
- ✓ разработка новых моделей автоматизируемых процессов с учетом проблем, диагностированных на предыдущем этапе;
- ✓ разработка модели ИС;
- ✓ разработка и внедрение полученной системы.

Разработка ИС идет в тесном сотрудничестве пользователей, аналитиков и разработчиков. Первые являются специалистами в предметной области. Вторые получают знания от первых и формулируют требования к разрабатываемой ИС. Разработчики реализуют полученные требования в виде готового продукта.

Далее рассмотрим некоторые этапы подробнее.

1.1.1. Организационно-технические системы

Под объектом автоматизации в работе рассматриваются организационно-технические системы, которые представляют собой совокупность организационной структуры и находящихся в ее распоряжении технических средств, т. е. совместно рассматривается человек и информационная система. Современные исследователи выделяют следующие особенности ОТС: многопараметричность, иерархичность, вероятностное поведение, сложность структуры и алгоритмов поведения.

Процессы, протекающие в ОТС, можно разделить на три группы:

- ✓ производственные и бизнес-процессы — процессы, связанные с основной деятельностью предприятия;
- ✓ процессы согласования;
- ✓ процессы принятия решений.

При автоматизации производственных и БП предприятия аналитики, опрашивая пользователей, получают информацию о структуре и функциях ОТС, строят модель БП и при необходимости дорабатывают ее. Таким образом, знания пользователя, обработанные аналитиком, преобразуются в техническое задание на разрабатываемую систему.

Для автоматизации процессов согласования лучше всего подходят мультиагентные системы.

Процессы принятия решений имеют свои особенности [9]. Прежде всего, данные задачи сложно описать алгоритмически. Решения принимаются по определенным сценариям, для описания которых целесообразно использовать базы знаний и технологии экспертных систем. При автоматизации деятельности лица, принимающего решение, возможно использование программных интеллектуальных агентов. Следовательно, для таких процессов ИС должна включать систему поддержки принятия решений, которая поможет ЛПР на основе имеющейся информации правильно определить проблему и выбрать оптимальное решение. Следует отметить, что не все алгоритмы и сценарии поведения поддаются полной формализации. В некоторых случаях требуется непосредственное участие ЛПР.

На рис. 1.1 показаны особенности автоматизации ОТС.

Автоматизация каждой группы процессов, протекающих в ОТС, имеет свои особенности. Существующие подходы к разработке, каждый в отдельности, не закрывают всех вопросов, возникающих при этом. Следовательно, целесообразно совмещать эти методы при разработке информационных систем.

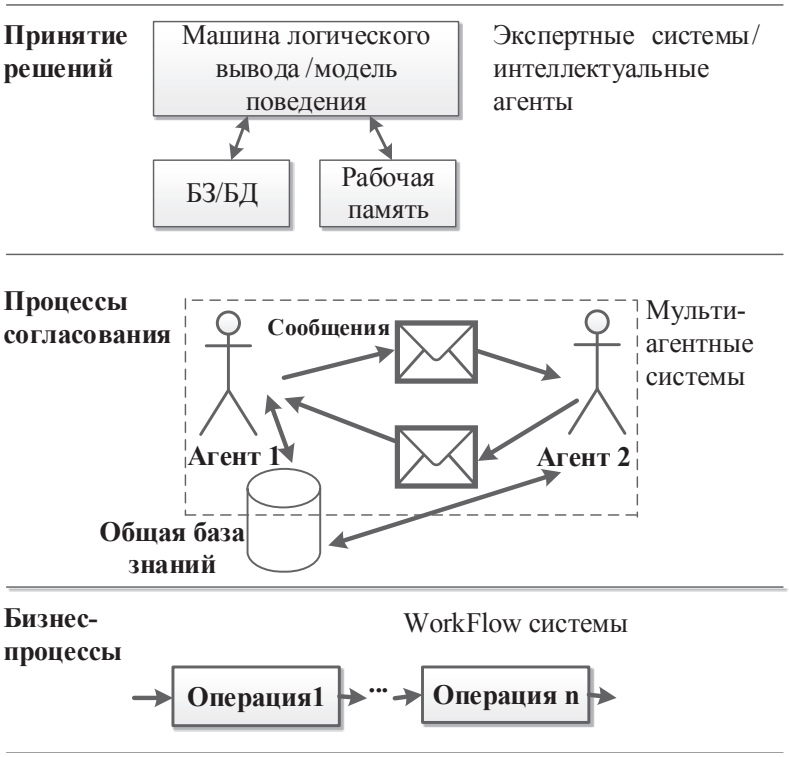

Рис. 1.1. Особенности автоматизации ОТС

1.1.2. Моделирование автоматизируемых процессов

Замещение одного объекта другим в целях получения информации о важнейших свойствах объекта-оригинала с помощью объекта-модели называется в учебниках моделированием. Для одной и той же системы можно построить несколько моделей, каждая из которых определяет конкретный аспект системы. При этом используются разные наборы диаграмм и документов.

Успешное решение задачи разработки информационной системы невозможно без четкого представления бизнес-процессов, протекающих на автоматизируемом предприятии. Поэтому, прежде всего, необходимо построить бизнес-модель, отражающую автоматизируемые процессы и необходимые для них ресурсы.

В процессе анализа предметной области при разработке информационной системы строятся модели деятельности предприятия. Они могут быть двух видов:

- ✓ модели «AS-IS» («Как есть»), которые отражают существующие бизнес-процессы предприятия;
- ✓ модели «AS-TO-BE» («Как будет»), которые показывают представления о новых процессах и технологиях работы предприятия.

Переход от модели «AS-IS» к модели «AS-TO-BE» происходит благодаря реинжинирингу (см. параграф 1.1.3). Далее на основе моделей «AS-TO-BE» строятся модели разрабатываемой ИС. Наличие моделей информационной системы также положительно сказывается на документировании проекта, поскольку принимаемые проектные решения становятся более наглядными. Состав моделей, используемых в каждом конкретном проекте, и степень их детализации зависят от следующих факторов [12]:

- ✓ сложность разрабатываемой системы;
- ✓ необходимая полнота ее описания;
- ✓ знания и навыки участников проекта;
- ✓ время, отведенное на разработку.

Поскольку проектирование и моделирование тесно связаны друг с другом, то при моделировании бизнес-процессов предприятия и структуры программного обеспечения также используют структурные и объектно-ориентированные методы.

1.1.2.1. Структурные методы анализа

Структурным анализом называется метод исследования системы, начинающийся с ее общего обзора, который затем детализируется, приобретая иерархическую структуру со все большим числом уровней [12]. Его основные характеристики:

- ✓ разбиение системы на уровни абстракции с ограничением числа элементов на каждом уровне;
- ✓ ограниченный контекст, включающий лишь существенные на каждом уровне детали;
- ✓ использование строгих формальных правил записи;
- ✓ последовательное приближение к конечному результату.

В структурном анализе и проектировании используют различные модели. Наиболее распространенными являются:

- ✓ функциональная модель SADT (IDEF0), которая описывает функциональную структуру системы;
- ✓ модель IDEF3, описывающая процессы, в которых важно понять последовательность выполнения действий и взаимозависимости

между ними, и часто использующаяся для детализации функциональных блоков IDEF0;
- ✓ диаграммы потоков данных DFD, которые описывают передачу информации между функциональными процессами;
- ✓ спецификация BPMN (англ. Business Process Model and Notation, нотация и модель бизнес-процессов) позволяет описать БП в виде последовательности объектов потока управления с возможностью задания циклических действий [5].

В семейство IDEF входят и другие модели, которые не так широко используются. Следует отметить методологию онтологического исследования сложных систем IDEF5. Она позволяет описать правила и ограничения, которые представляют состояние системы, с использованием словаря терминов [13].

Можно выделить следующие положительные черты структурных методов:

- ✓ графические диаграммы позволяют наглядно представить структуру системы;
- ✓ визуальное моделирование доступно неспециалистам в области информационных технологий;
- ✓ иерархическая структура диаграмм позволяет описывать функции системы с разной степенью подробности.

При разработке ИС целесообразно использовать диаграммы потоков данных DFD [12], поскольку этот метод хорошо согласуется со средством моделирования данных (модель «сущность-связь»). Хранилища данных, описываемые на диаграммах DFD, являются основой для построения модели «сущность-связь».

1.1.2.2. Объектно-ориентированные методы анализа и проектирования ПО

IDEF4 — методология построения объектно-ориентированных систем. Средства IDEF4 позволяют наглядно отображать структуру объектов и заложенные принципы их взаимодействия. Это позволяет анализировать и оптимизировать сложные объектно-ориентированные системы.

Язык UML — широко используемый в настоящее время метод объектно-ориентированного анализа [12]. Язык UML (версия 2.0) представляет собой набор из 13 диаграмм [7], которые позволяют описать разрабатываемую информационную систему с двух сторон: логиче-

ском/физическом и статическом/динамическом. Для описания статических частей системы используются следующие диаграммы: классов, объектов, компонентов и развертывания. Для описания динамической составляющей ИС используются диаграммы прецедентов, последовательности, кооперации, состояний и деятельности.

Широкое применение языка UML во многом связано с возможностью расширения его существующих элементов, что позволяет создавать новые языки для использования в конкретной предметной области на основе UML. В своей работе авторы используют следующие диаграммы:

- ✓ вариантов использования (предназначены для определения множества функций, которые будет выполнять ИС);
- ✓ последовательности (объекты ИС взаимодействуют между собой путем передачи сообщений друг другу; эти диаграммы позволяют описать последовательность передачи сообщений между объектами);
- ✓ классов (позволяют создавать логическое представление системы, т. е. описать иерархию классов, а также методы и их свойства).

Моделирование БП является важной составляющей разработки ИС. Существуют структурные и объектно-ориентированные методы анализа и проектирования ПО. Каждый из них имеет свои достоинства и недостатки. Для получения наибольшего эффекта ставится и решается задача интеграции этих методов для задачи разработки ИС.

1.1.3. Реинжиниринг бизнес-процессов

Современное предприятие имеет довольно сложную структуру и алгоритмы работы. На начальном этапе разработки ИС необходимо удостовериться в том, что организация существующих БП позволяет работать предприятию оптимально, и если это не так, то внести соответствующие изменения. Для этого используется реинжиниринг бизнес-процессов — фундаментальное переосмысление и радикальное перепроектирование БП для достижения коренных улучшений в основных показателях деятельности предприятия [22]. Целью РБП является системная реорганизация всех потоков предприятия (материальных, финансовых и информационных), которая должна привести к упрощению организационной структуры, перераспределению и минимизации использования различных ресурсов, сокращению сроков

реализации потребностей клиентов. Все это в конечном итоге позволит повысить качество работы предприятия [18].

Наиболее распространенным средством моделирования динамических процессов (переходов из одного состояния в другое, из одной ситуации в другую) является имитационное моделирование и, в частности, дискретно-событийное.

Также в имитационном моделировании используется мультиагентный подход. При этом поведение системы определяется как результат деятельности многих агентов. Для построения мультиагентной модели необходимо определить поведение отдельных агентов и взаимоотношения между ними.

Для анализа, совершенствования и реинжиниринга БП в ОТС используются средства имитационного и мультиагентного моделирования, что повышает эффективность автоматизации.

Итак, были рассмотрены этапы разработки информационной системы, предшествующие написанию технического задания. Этот документ является связующим звеном между участниками ИТ-проекта. От качества описания каждой группы процессов, протекающих в ОТС, зависит результат разработки информационной системы. Решение задачи автоматизации совместно с задачей совершенствования БП позволяет повысить эффект от внедрения ИС.

1.2. Риски, связанные с разработкой информационных систем, и пути их снижения

Разработка ИС сопровождается определенными рисками [11]. Перечислим некоторые из них:

1. Сроки выполнения этапов могут отставать от графика проекта или бюджет проекта будет существенно превышен, что может привести к закрытию проекта до завершения, следовательно, ИС даже не будет разработана.

2. Низкое качество внедренной ИС может привести к отказу от ее использования.

3. Разработанная ИС теряет свою полезность после нескольких лет эксплуатации, поскольку количество недостатков и стоимость внесения изменений увеличивается настолько, что становится проще и дешевле разработать новую систему, чем поддерживать существующую.

4. В процессе эксплуатации ИС выясняется, что она не решает необходимых задач предприятия, а это может быть следствием изначально неточной постановки задачи или изменений БП в ходе разработки ИС, которые не учитывались.

5. В процессе эксплуатации ИС выясняется, что она имеет множество функций, не используемых заказчиком, а действительно полезные в системе не реализованы.

6. В течение жизненного цикла ИС происходит почти полное обновление команды разработчиков. Недостаточное документирование системы не позволяет новым сотрудникам успешно модернизировать ИС.

Проекты, длящиеся более двух лет, являются довольно рискованными [1]. На рис. 1.2 показан процент успешных проектов в зависимости от их длительности и характеристик (завершенность в срок, уложились в бюджет и т. п.).

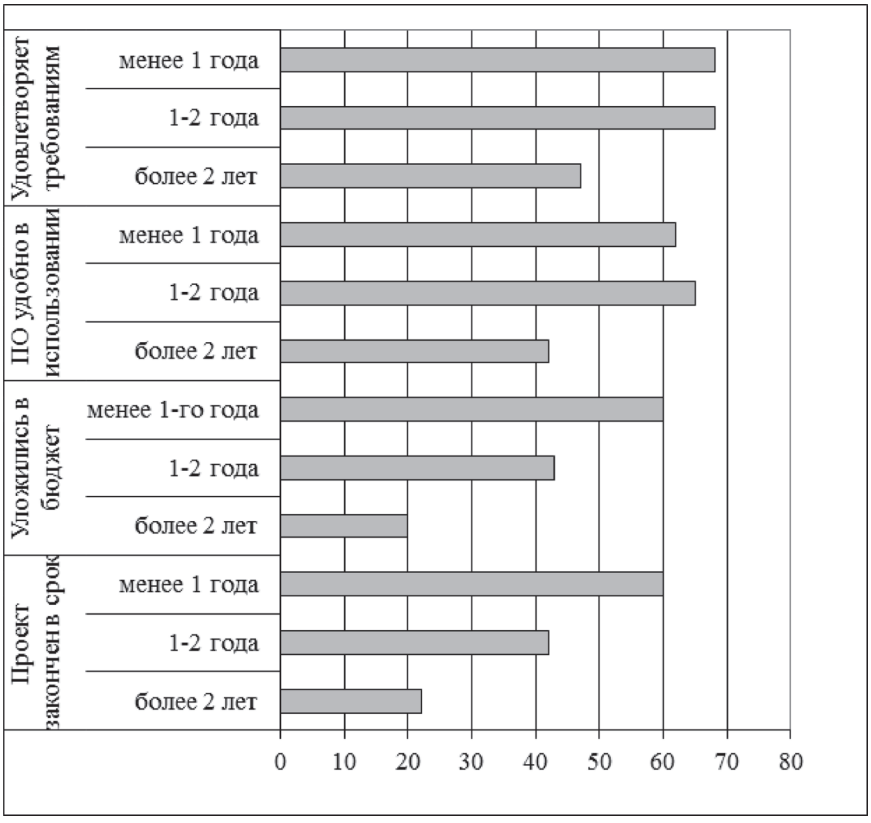

Рис. 1.2. Процент успешных проектов в зависимости от их длительности и характеристик

В настоящее время создание ИС рассматривается как формализованный процесс, который должен соответствовать определенным стандартам и нормативным документам [12]. Создаются различные методы и технологии разработки ИС, которые позволяют снизить риски, связанные с ее разработкой, увеличить производительность участников ИТ-проекта, улучшить качество разрабатываемой системы за счет повышения управляемости процесса создания ИС и своевременного учета изменившихся требований к системе.

Следует отметить, что эффективность применения этих методов и технологий напрямую связана с наличием программного средства, обеспечивающего автоматизацию этапов — CASE-сродство (Computer Aided Software Engineering — компьютерно-ориентированная программная инженерия).

Метод разработки включает в себя:

- ✓ определенную последовательность этапов разработки;
- ✓ перечень нотаций (графических и текстовых средств), используемых для описания создаваемой системы;
- ✓ критериев и правил, используемых для оценки полученных результатов.

Таким образом, применение определенных методов разработки ПО позволяет уменьшить возможные риски, а наличие CASE-средств, которые их поддерживают, — повысить эффективность использования метода разработки программного обеспечения.

1.3. Методологические и теоретические основы поддержки принятия решений, моделирования и разработки информационных систем

1.3.1. Применение имитационного моделирования при разработке программного обеспечения

Принятие решения о внедрении на предприятии единой ИС связано со многими трудностями. Можно выделить следующие основные проблемы, возникающие при этом:

- ✓ наличие нескольких несвязанных ИС, решающих узкие проблемы;
- ✓ неоптимальные БП, которые не позволяют эффективно работать предприятию;

- ✓ сложность прогнозирования характеристик работы ИС (скорость обработки запросов к базе данных, скорость передачи данных по сети и т. п.).

Для решения первой проблемы необходимо спроектировать новую БД для единой ИС и перенести в нее накопленные данные.

Наиболее эффективным путем решения второй проблемы является построение модели существующих БП и проведение их РБП. Имитационное моделирование на этом этапе существенно помогает в анализе такой сложной системы, как современное предприятие.

Имитационная модель описывает законы функционирования каждого элемента системы и их взаимосвязи. В результате имитационного эксперимента по исходным данным можно получить сведения о состоянии БП в определенные моменты времени. Это позволяет оценить характеристики системы и лучше понять, как разработать систему, удовлетворяющую заданным критериям оценки эффективности. Для получения достоверных и точных результатов ИМ при небольших затратах машинного времени необходимо осуществлять планирование экспериментов.

Для прогнозирования характеристик работы ИС можно расширить имитационную модель так, чтобы она включала в себя не только описание автоматизируемых БП, их статистические характеристики, такие как время обработки заявки (документа), время простоя в очереди, количественные характеристики процессов, а также временные и количественные характеристики транзакций в ИС. Эксперименты с такой имитационной моделью позволят до реализации готовой ИС получить представление о ее производительности, требования к скорости и интенсивности обработки заявок (документов) и транзакций в БД.

Применение ИМ при разработке ИС целесообразно как на этапе анализа автоматизируемых БП в целях их оптимизации, так и перед разработкой ИС для прогнозирования характеристик ее работы.

1.3.2. Использование систем искусственного интеллекта при разработке информационных систем

Использование искусственного интеллекта при разработке ИС переводит этот процесс на качественно новый уровень, поскольку позволяет автоматизировать не только рутинные операции, но и процесс принятия решений при создании ИС. Для этого инструментарий, под-

держивающий процесс разработки, должен включать в себя ЭС. Поскольку невозможно полностью исключить человека из процесса разработки ИС, то современные исследователи предлагают использовать диалоговые ЭС.

В БЗ интеллектуальной СППР в области разработки ПО (CASE-средстве) должны храниться факты и правила, полученные от экспертов предметной области, а также профессиональные знания разработчиков. Использование такой СППР позволит решать практические задачи, возникающие в процессе разработки ИС.

Развитие информационных технологий определило повышенное внимание к интегрированному использованию ИС и ЭС в части автоматизации деятельности ЛПР. Такие системы могут быть использованы в задачах, для решения которых требуется взаимодействие целого ряда экспертов предметной области. Использование БЗ и машины логического вывода в ИС позволят автоматизировать поиск решения, к которому мог бы прийти в аналогичной ситуации эксперт.

Учет большого объема данных, знаний экспертов-специалистов, формирование ответа на основании некоторого аналога процесса рассуждений, диалог с пользователем — все это делает ЭС тем инструментом, который позволит сделать более интеллектуальным процесс создания ИС и автоматизировать работу ЛПР.

1.3.3. Применение мультиагентного подхода

Мультиагентные системы представляют собой новое направление развития искусственного интеллекта, которое сформировалось на основе результатов исследований современных ученых в области распределенных компьютерных систем, сетевых технологий решения проблем и параллельных вычислений.

Агентно-ориентированный подход уже используется при распределенном решении сложных задач, реинжиниринге предприятий, электронном бизнесе, логистике, а также моделировании.

Агент — это аппаратная или программная сущность, способная действовать в интересах достижения целей, поставленных перед ней владельцем и (или) пользователем [8]. Агенты описываются также рядом свойств, которые характеризуют понятие агента. Обычно агент обладает набором следующих свойств: адаптивность, автономность, сотрудничество, способность к рассуждениям, коммуникативность, мобильность.

Следует отметить, что агенты могут быть отнесены к следующим типам: реактивные, интеллектуальные, гибридные.

Реактивный агент принимают решения на основе знаний «ситуация-действие». Интеллектуальный агент решает поставленные перед ним задачи, исходя из своих целей и используя общие ограниченные ресурсы и знания о внешнем мире. Гибридный агент сочетает в себе возможности первых двух.

Интеллектуальная МАС представляет собой множество интеллектуальных агентов, распределенных в сети. Они отслеживают необходимые данные и взаимодействуют друг с другом для достижения поставленных перед ними целей. Можно выделить несколько важных причин взаимодействия агентов: совместимость целей (общая цель), отношение к ресурсам, необходимость привлечения недостающего опыта, взаимные обязательства.

С точки зрения программирования, агенты представляют собой ПО, которое способно действовать самостоятельно от лица пользователя. При создании МАС может использоваться архитектура «клиент-сервер».

Введение программных агентов в ИС позволит в некоторой степени упростить работу пользователей, поскольку агенты смогут отслеживать состояния системы и предлагать определенные решения.

Важной проблемой при разработке ИС является получение знаний экспертов в определенной предметной области. Эти знания большей частью не формализованы, а поэтому недоступны другим людям. Удачное решение этой проблемы — добавление в создаваемую ИС общей БЗ и разработка агентов, которые на основе этих знаний будут предлагать решения определенных проблем в автоматизируемой области. Таким образом, проектируемая ИС расширяется до интеллектуальной СППР, которая выполняет определенные формализованные функции пользователей и оказывает поддержку при решении задач организационно-технического управления.

Также мультиагентный подход может быть применен при разработке распределенных ИС. В этом случае каждый узел системы представляет собой программного агента со своими ресурсами. Их кооперация обеспечивает функционирование системы в целом.

Перечислим следующие требования к методу разработки ИС:

- ✓ возможность разработки модели лица, принимающего решения, представленного интеллектуальным агентом;
- ✓ возможность описания сценариев принятия решения ИА.

1.4. Обзор и сравнительный анализ существующих систем поддержки принятия решений в области разработки информационных систем (CASE-средств)

1.4.1. Классификация CASE-средств

В современные CASE-средствах реализована поддержка разнообразных технологий проектирования ИС. К ним можно отнести простые средства анализа и документирования, а также полномасштабные средства автоматизации, охватывающие весь жизненный цикл ПО.

Этапы анализа и проектирования являются наиболее трудоемкими при разработке ИС, поэтому использование CASE-средств позволяет повысить качество принимаемых технических решений и облегчить подготовку проектной документации.

На рынке программных средств в настоящее время представлено большое количество различных CASE-средств. Одни из них — достаточно дешевые системы для персональных компьютеров с весьма ограниченными возможностями, другие — мощные дорогостоящие системы, поддерживающие процесс разработки ПО, начиная от формулирования требований и заканчивая внедрением.

По мнению современных авторов, к CASE-средствам относят любое программное средство, автоматизирующее ту или иную совокупность процессов жизненного цикла ПО и обладающее следующими основными характерными особенностями:

- ✓ мощные графические средства для описания и документирования ИС, обеспечивающие удобный интерфейс с разработчиком и развивающие его творческие возможности;
- ✓ интеграция отдельных компонент CASE-средств, обеспечивающая управляемость процессом разработки ИС;
- ✓ использование специальным образом организованного хранилища проектных метаданных.

Наиболее используемыми CASE-средствами являются CA Erwin Modeling Suite, Rational Suite, ARIS Toolset, Power Designer и Borland Together Designer, BizAgi BPM Suit, ELMA BPM (в следующем подразделе приводится их описание).

1.4.2. Описание CASE-средств

1.4.2.1. CA ERwin Modeling Suite

CA Erwin Modeling Suite (URL: http://www.interface.ru) — комплект инструментов фирмы CA Technologies, которые в полной мере обеспечивают решение всех задач анализа, проектирования, генерации, тестирования и сопровождения информационных систем. Кратко опишем продукты, входящие в этот пакет.

Process Modeler (ранее BPwin) — инструмент визуального моделирования бизнес-процессов. Дает возможность представить любую деятельность или структуру в виде модели. Process Modeler поддерживает три нотации моделирования: IDEF0, IDEF3 и DFD.

Erwin Data Modeler (ERwin) позволяет проектировать, документировать и сопровождать БД и хранилища данных. ERwin поддерживает следующие СУБД: СУБД DB2 для LUW и z/OS, СУБД SQL Server 2008 и СУБД Teradata 13.0, Oracle, Interbase, Ingres.

Component Modeler (Paradigm Plus) — средство для моделирования компонентов ПО и генерации объектного кода приложений на основе созданных моделей. Интегрирован с Process Modeler, что дает дополнительные возможности при работе с функциональными моделями. Component Modeler обеспечивает полную поддержку UML, поддерживает синхронизацию проектирования и реализации приложения. Component Modeler поддерживает генерацию программного кода и обратный инжиниринг программных сборок Microsoft.NET (C# и Visual Basic), Microsoft Visual J++ и Microsoft Visual C++.

Таким образом, CA Erwin Modeling Suite позволяет описывать модель БП в нотациях IDEF0, IDEF3, DFD, разрабатывать модель данных и архитектуру ПО с поддержкой различных СУБД. Для анализа и оптимизации БП предлагается использовать функционально-стоимостной анализ. В настоящее время данный пакет особое внимание уделяет моделированию.

1.4.2.2. Продукты компании IBM Rational

Продукты компании IBM Rational (URL: http://www.interface.ru) представляет собой комплекс инструментальных средств, поддерживающих весь жизненный цикл разработки ПО, и интегрирован с IBM

Rational Unified Process (RUP). Технически RUP оформлена в виде размещенной на Web базы знаний, которая снабжена поисковой системой.

CASE-средство IBM Rational Software Architect предоставляет разработчику следующие возможности: анализ и проектирование ПО с помощью UML, анализ и контроль структуры приложений Java, подготовка проектной документации. Его компоненты основаны на технологии Eclipse. Расширение Simulation позволяет провести имитационное моделирование поведения ПО на основе UML-диаграмм деятельности, последовательности, коммуникации или диаграммы состояний. Rational Software Architect поддерживает генерацию программного кода и обратный инжиниринг: Microsoft Visual Studio 2010 и.NET Framework 4 для C# и VB. NET, Java и C++.

Технология создания ПО, реализованная в IBM Rational, также может применяться для проектирования пользовательского интерфейса. Разработка ПИ основана на концепции сценариев вариантов использования (use-case story board) и представляет собой модель взаимодействия пользователя с ПИ. Такой подход позволяет качественно оценить предложенные решения прежде, чем реальный интерфейс будет спроектирован и реализован.

IBM Rational Rose Data Modeler представляет собой средство моделирования данных в нотации ER.

Продукты компании IBM Rational уделяют внимание разработке архитектуры организации и программного обеспечения, управлению жизненным циклом созданной информационной системы, не затрагивая вопросов, связанных с моделированием данных.

ARIS Toolset

Интегрированная среда ARIS Toolset (URL: http://consulting.ru) представляет собой комплекс инструментов:

- ✓ для проектирования и управления предприятием;
- ✓ моделирования, анализа и оценки бизнес-процессов;
- ✓ документирования бизнес-процессов в соответствии с требованиями международных стандартов;
- ✓ разработки, внедрения и сопровождения ИС.

Методология моделирования и проектирования ИС, реализованная в этой системе, основывается на совокупности различных методов моделирования, отражающих разные взгляды на проектируемую систему (организация, функции и цели, данные, продукты и услуги, процессы). При построении моделей в ARIS Toolset можно использо-

вать как собственные методы моделирования ARIS, так и известные языки моделирования, например UML, BPMN.

Интегрированная среда ARIS Toolset занимается имитационным моделированием БП, их анализом, совершенствованием и оптимизацией, есть возможность использовать нотацию ER для разработки модели данных. Вопросы, касающиеся разработки архитектуры ПО, решаются с помощью других приложений, с которыми у ARIS есть интеграция.

Power Designer

Power Designer (URL: http://www.interface.ru) представляет собой CASE-средство фирмы Sybase и имеет модульную архитектуру. Power Designer имеет следующие возможности:

- ✓ структурное моделирование бизнес-процессов;
- ✓ концептуальное и физическое проектирование и генерация БД (поддерживает более 60 СУБД);
- ✓ объектно-ориентированный анализ и моделирование данных с использованием UML;
- ✓ интеграция с ведущими средами разработки (Eclipse, Microsoft Visual Studio, Power Builder).

Power Designer поддерживает моделирование данных, статическое моделирование БП и приложений. Напрямую не затрагивает следующие вопросы: управление требованиями и конфигурациями ПО; имитационное моделирование.

Borland Together 2008

Borland Together (URL: http://www.interface.ru) представляет собой пакет программ фирмы Micro Focus/Borland. Он предназначен для создания моделей, представляющих собой схему бизнес-процессов, структуру данных и архитектуру приложений и предприятия. Borland Together состоит из следующих программ: Borland Together Designer, Borland Together Designer Community Edition и Borland Together Developer.

Borland Together предоставляет следующие возможности:

- ✓ визуальное моделирование метамоделей для определенной предметной области;
- ✓ моделирование бизнес-процессов с использованием нотаций Business Process Modeling, BPMN;
- ✓ разработка проектов графических моделей программных приложений в нотации языка UML;

- ✓ разработка логических диаграмм данных «сущность-связь» в нотации ER и IDEF1x;
- ✓ прямое и обратное проектирование для ведущих СУБД (Oracle, DB2, Sybase, MS SQL Server);
- ✓ генерирование документации;
- ✓ генерирование программного кода для Java, C++ и C#;
- ✓ распознавание шаблонов проекта исходного кода.

Borland Together 2008 является CASE-средством, которое автоматизирует процесс разработки ПО на этапах моделирования БП, создания диаграмм «сущность-связь» с возможностью генерации структуры БП, моделирования ПО, заканчивающееся генерированием программного кода.

BizAgi BPM Suit

Система BizAgi BPM Suit (URL: http://www.b-k.ru/products/bizagi) представляет собой платформу для автоматизации БП. Для моделирования БП используется приложение BizAgi Process Modeler, которое использует нотацию BPMN. BizAgi Studio позволяет преобразовать описанный БП в работающее приложение без участия программиста. Модель находится в хранилище сервера, интерпретируется и выполняется через Web-приложение на BizAgi BPM Server. Этот сервер позволяет анализировать различные показатели выполнения БП в целях выявления проблемных мест и их улучшения.

Важно сделать замечания относительно функционала приложений, разрабатываемых с помощью BizAgi, и моделируемых процессов: 1) функциональность приложений относится к классу Workflow; 2) моделирование касается только организационных процессов.

Система BizAgi BPM Suit относится к классу систем управления БП и позволяет разрабатывать проблемно-ориентированные приложения класса WorkFlow, с возможностями аналитики и контроля. Данная система не поддерживает проектирование архитектуры более широкого класса ПО.

ELMA BPM Suite

ELMA BPM Suite (URL: http://crm74.ru) включает в себя приложение для управления показателями БП и управление БП. В программе «Дизайнер ELMA» проводится моделирование БП с использованием нотации BPMN. Спроектированная модель хранится на сервере приложения. Далее происходит выполнение БП, причем можно запускать

на исполнение несколько экземпляров одного и того же БП. Пользователь получает информацию о задачах, которые ему необходимо выполнить, в виде страниц в браузере. Система автоматически формирует задачи для каждого сотрудника при выполнении БП. С помощью специальных приложений осуществляется мониторинг и контроль БП.

Система управления БП ELMA BPM Suite позволяет разработать внутренний портал предприятия для управления его работой. Кроме того, предлагаются готовые решения для электронного документооборота, управления проектами. ELMA не поддерживает UML и проектирование широкого класса ПО, имитационное моделирование процессов.

1.4.3. Критерии сравнения функциональных возможностей CASE-средств

Для сравнительной оценки функциональных возможностей CASE-средств предлагается следующий набор критериев:

- ✓ нотации, используемые при описании бизнес-процессов и архитектуры ИС (IDEF0, DFD, UML, BPMN);
- ✓ использование имитационного моделирования для анализа, совершенствования и оптимизации БП;
- ✓ проектирование ПО на основе имитационной модели БП. Данная возможность позволяет использовать информацию из модели при проектировании ПО, что снижает трудоемкость работы;
- ✓ разработка моделей лиц, принимающих решения;
- ✓ описания сценариев принятия решений ЛПР;
- ✓ проектирование пользовательского интерфейса, разработка макета (предварительного прототипа) пользовательского интерфейса;
- ✓ конвертация одних диаграмм в другие, при этом для построения новых диаграмм используется информация, уже отраженная в первых диаграммах;
- ✓ проектирование структуры БД;
- ✓ возможность генерации исполняемого кода;
- ✓ генерация технической документации к проектируемой ИС. Поскольку информация по проекту хранится в единой БД и автоматически обновляется при внесении в него изменений, то в любой момент имеется возможность автоматически создавать актуальную техническую документацию.

1.4.4. Сравнительный анализ CASE-средств

Ниже приводится сравнение программных продуктов ARIS Toolset, Power Designer, Borland Together Designer, Продукты IBM Rational, CA ERwin Modeling Suite, BizAgi и Elma в области проектирования ПО (см. таблицу). Они являются наиболее распространенными среди пользователей — разработчиков программных систем.

Сравнение CASE-средств

Критерии сравнения	Продукты IBM Rational	CA ERwin Modeling Suite	ARIS Toolset	Power Designer	Borland Together Designer	BizAgi	Elma
Поддержка IDEF0, DFD	Нет	Да	Да	Нет	Нет	Нет	Нет
Поддержка UML	Да	Да	Да	Да	Да	Нет	Нет
Поддержка BPMN	Нет	Нет	Да	Нет	Да	Да	Да
ИМ БП	Нет	Нет	Да	Нет	Нет	Да	Нет
Проектирование ПО на основе ИМ БП	Нет	Нет	Нет	Нет	Нет	Да	Да
Разработка модели ЛПР	Нет	Нет	Нет	Нет	Нет	Да	Да
Описание сценария принятия решений ЛПР	Нет	Нет	Нет	Нет	Нет	Да	Нет
Проектирование ПИ	Нет	Нет	Нет	Нет	Нет	Да	Нет
Конвертация диаграмм	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Проектирование структуры БД	Да	Да	Да	Да	Да	Нет	Нет
Генерация техдокументации к создаваемой ИС	Да	Да	Да	Да	Да	Да	Нет

В пакете CA ERwin Modeling Suite отсутствуют средства имитационного моделирования бизнес-процессов, но есть возможность экспортировать модель в систему моделирования Arena.

ARIS Toolset, помимо указанных нотаций, использует большое количество других нотаций (диаграммы Чена, Object Modeling Technique).

Следует отметить, что большое количество нотаций требует дополнительных знаний от аналитика для их использования. В пакете ARIS Toolset имеется адаптированный движок моделирования, к недостаткам которого можно отнести отсутствие поддержки моделирования систем массового обслуживания.

Power Designer и Borland Together Designer прежде всего являются средствами для разработки UML-диаграмм и генерирования программного кода, они не предоставляют возможностей для анализа и имитационного моделирования процессов. Рассмотренные системы класса BPM подходят для разработки узкого класса систем.

Все рассмотренные продукты можно разделить на четыре группы:

- ✓ системы, охватывающие весь жизненный цикл программного обеспечения (продукты IBM Rational, CA ERwin Modeling Suite);
- ✓ системы, автоматизирующие разработку программного обеспечения (Power Designer и Borland Together Designer);
- ✓ системы анализа и имитационного моделирования бизнес-процессов (ARIS Toolset);
- ✓ системы управления бизнес-процессами (BizAgi и Elma).

Поскольку в работе рассматриваются задачи автоматизации процессов организационно-технических систем, то наилучшими вариантами представляются системы третьей группы (ARIS Toolset).

К недостаткам описанных CASE-средств можно отнести следующее:

- ✓ отсутствует интеграция структурного и объектно-ориентированного подхода;
- ✓ отсутствие интеллектуальности процесса проектирования (не решена задача автоматического перехода к проектированию одних диаграмм на основе других);
- ✓ управление бизнес-процессами только определенного класса систем (Workflow);
- ✓ отсутствие пакетов для имитационного моделирования бизнес-процессов, а следовательно, и возможности использовать информацию из модели при проектировании информационной системы.

В следующей главе описывается новый метод поддержки принятия решений при разработке информационных систем и CASE-средство, которые устранили эти недостатки.

2. Метод поддержки принятия решений при разработке информационных систем для предметной области мультиагентных процессов преобразования ресурсов

2.1. Требования к модели и методу поддержки принятия решений при разработке информационных систем

Рассмотрим разработку ИС предметной области мультиагентных процессов преобразования ресурсов. На основе проведенного анализа методологических и теоретических основ поддержки принятия решений, моделирования и разработки ИС (см. подглаву 1.3) и CASE-средств (см. параграф 1.4.4) сформулируем требования к методу поддержки принятия решений в области разработки ИС.

Метод ППР при разработке ИС должен обеспечивать автоматизацию процесса создания ИС для предметной области организационно-технических систем. Таким образом, можно выделить следующие требования:

- ✓ выбор методики системного анализа и модели для формализации процессов ОТС. При этом будем учитывать наличие ЛПР, которые могут быть представлены в виде ИА;
- ✓ ИМ для проверки модели «Как будет» на этапе реинжиниринга БП и оценки производительности ИС;
- ✓ интеллектуальная разработка ИС, включающая функциональный и объектно-ориентированный анализ, моделирование ПИ, формирование исполняемого кода ИС.

Рассмотрим принципы построения интеллектуальных СППР в области разработки ПО (CASE-средств). Структура такой системы включает в себя диалоговую ЭС, состоящую из базы знаний, БД, механизма логического вывода, а также блок объяснения полученных решении, блок обучения (адаптация ЭС к изменяющейся действительности), блок понимания, блок ведения, пополнения и корректировки БЗ.

В подглаве 2.2 проведено сравнение различных моделей БП и выбрана одна для представления ППР. Задача выбора модели представления знаний о предметной области решается в подглаве 2.3.

2.2. Выбор модели представления бизнес-процессов

Математические модели дискретных процессов для представления ППР: сети Петри, расширенные сети Петри, системы массового обслуживания, модели системной динамики — не обеспечивают всех требований для моделирования ППР. Поэтому математическая модель ППР была расширена аппаратом мультиагентных систем [9].

Динамическая модель МППР [10] включает процессы (*PR)*, операции (*Op*), ресурсы (*RES*), команды управления (*U*), средства (*MECH*), источники (*Sender*) и приемники ресурсов (*Receiver*), перекрестки (*Junction*), параметры (*P*), агенты (*Agent*). Отдельно выделены информационные типы ресурсов: сообщения (*Message*) и заявки на выполнение операции (*Order*). Описание причинно-следственных связей между элементами преобразования и ресурсами задается объектом «связь» (*Relation*).

i-я операция (Op_i) представлена следующей структурой:

$Op_i = <f, RESin_i, RESout_i, MECH_i>$,

где *f* — функция, которую реализует *i*-я операция;

$RESin_i$ — входные ресурсы для выполнения *i*-й операции;

$RESout_i$ — выходные ресурсы для выполнения *i*-й операции:

$RESout_i = f(RESin_i)$;

$MECH_i$ —механизмы для выполнения *i*-й операции.

Здесь $RESin_i = \{RESin_{i1}, RESin_{i2}, ..., RESin_{ik}\}$ — множество входных ресурсов;

$RESout_i = \{RESout_{i1}, RESout_{i2}, ..., RESout_{im}\}$ — множество выходных ресурсов.

Семантика процесса преобразования ресурса показана на рис. 2.1.

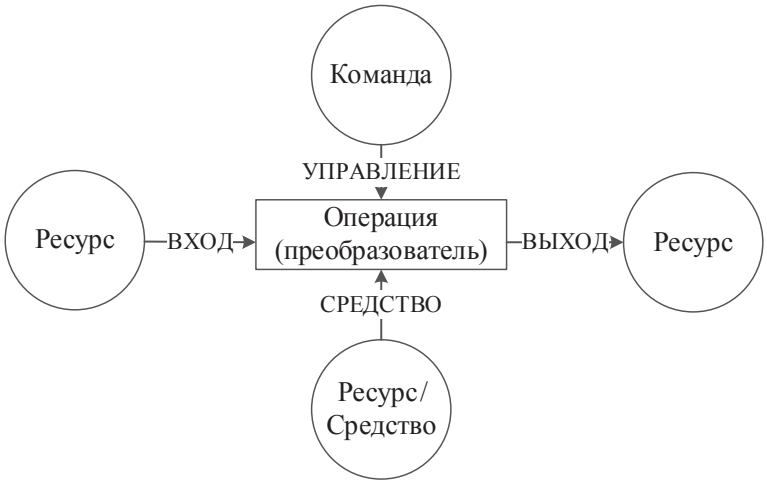

Рис. 2.1. Семантика процесса преобразования ресурса

Результаты сравнения математических моделей дискретных процессов приведены в табл. 2.1.

Таблица 2.1

Результаты сравнения математических моделей дискретных процессов

Характеристики	Сети Петри	Расширенные сети Петри	СМО	Модели системной динамики	Модель МППР
Учет временных характеристик	Нет	Да	Да	Да	Да
Возможность учета различных типов ресурсов	Нет	Да	Да	Нет	Да
Моделирование конфликтов на общих средствах	Нет	Нет	Да	Нет	Да
Модель ЛПР (ИА)	Нет	Нет	Нет	Нет	Да

Полученную математическую модель МППР предлагается использовать в качестве модели описания БП. К преимуществу модели МППР относится возможность использования моделей ЛПР.

При разработке CASE-средств важно выбрать модель представления знаний (для реализации семантики перехода от объектов автоматизируемого процесса к объектам предметной области ИС). Эта задача решается в подглаве 2.4.

2.3. Выбор модели представления знаний

Рассмотрим существующие в настоящее время в литературе способы представления знаний. При анализе учитываются следующие требования:

- ✓ наиболее простой и естественный переход от неформализованных знаний и представлений к формальным моделям для описания БП и ИС, наглядность представления информации для пользователя;
- ✓ удобство представления иерархических данных, поскольку предметные области МППР и ИС образуют иерархию;
- ✓ простота добавления новых знаний;
- ✓ техническая реализация выбранной модели должна быть достаточно простой и согласовываться с объектно-ориентированным подходом разработки программного обеспечения;
- ✓ возможность использования языка UML в качестве визуального языка представления знаний (объектно-ориентированный анализ).

1-й способ. Представление знаний в виде продукций или правил и их интерпретатор, определяющий когда и какое правило применяется

В продукционную модель достаточно просто добавлять новые знания, поскольку любая продукция может размещаться в любом месте модели. Механизм вывода в этом способе хорошо сочетается с процедурным подходом в программировании.

В качестве недостатков можно указать следующее: неудобно реализовывать иерархическую структуру, ненаглядное представление знаний и неэффективный процесс вывода, поскольку в общем случае необходимо проверить применимость всех правил.

2-й способ. Представление знаний в виде семантической сети

В литературе дается следующее определение семантической сети. Семантическая сеть — это ориентированная графовая структура, каждая вершина которой отображает некоторое понятие (объект, процесс, ситуацию), а ребра графа соответствуют отношениям типа «это есть», «принадлежать», «быть причиной», «входить в», «состоять из», «быть как» и аналогичным между парами понятий.

Можно указать следующие достоинства данного способа: наглядность представления знаний для пользователей, семантическая модель

хорошо сочетается с иерархическими знаниями. К недостаткам можно отнести сложность реализации механизма вывода.

3-й способ. Представление знаний в виде фреймов

Определение фрейма дал М. А. Минский. Он определил фрейм как структуру данных для представления стереотипных ситуаций [2]. Фреймовая модель представления знаний задает основу описания класса объектов и удобна для описания структуры и характеристик однотипных объектов (процессов, событий), описываемых фреймами — специальными ячейками (шаблонами понятий) фреймовой сети (знания).

Фрейм состоит из слотов, которые представляют собой различные характеристики объекта (атрибуты), и наполнителей (значения этих атрибутов и процедуры, выполняющиеся при изменении данных фрейма). У каждого фрейма есть специальный слот, который хранит наименование сущности, которую он представляет. Также, фрейм можно рассматривать как узел некоторой сети. Связи могут быть следующих типов: экземпляр-класс и класс-суперкласс.

Фреймовое представление знаний достаточно наглядно подходит для представления иерархических знаний, хорошо сочетается с объектно-ориентированным подходом, поэтому переход от описания знаний к программной реализации информационной системы происходит достаточно просто. К недостаткам фреймовой модели можно отнести сложность внесения изменений в иерархическую структуру данных.

Сочетание семантических сетей и фреймов позволяет минимизировать недостатки этих двух способов с сохранением достоинств.

4-й способ. Применение фреймово-семантической модели представления знаний

Швецов А. Н. [23] предложил совместить фреймоподобные структуры с конструкциями концептуальных графов J. F. Sowa [4]. Фрейм-концепт состоит из следующих частей:

- ✓ имя фрейма является уникальным идентификатором, позволяющим однозначно определить фрейм;
- ✓ информация о применении фрейм-концепта представляет собой описание возможных ситуаций его использования, сценариев поведения, особенностей выбора в произвольной форме;
- ✓ структура сценариев поведения, которая описывает динамическое поведение компонентов или агентов предметной области

и в которую включен блок выбора сценария, позволяющий формировать альтернативные пути поведения данного фрейма;

- ✓ структура слотов состоит из структуры концептов и структуры атрибутов.

Для установления логической организации предметной области фрейм-концепты соединяются в структуры концептуальных графов. Концептуальный граф — это двудольный граф, имеющий два типа вершин: вершины концептов, или концептуальные вершины, и вершины концептуальных отношений.

Достоинства фреймово-семантического подхода: эффективно реализует иерархическое представление данных, хорошо сочетается с объектно-ориентированным подходом. В настоящее время уже решена задача технической реализации данной модели представления знаний на уровне реляционной базы данных [9].

Сравнительный анализ рассмотренных моделей представления знаний показан в виде табл. 2.2.

Таблица 2.2

Сравнение различных моделей представления знаний

Требования к модели	Продукции	Семантические сети	Фреймы	ФК и КГ Швецова
Наглядность	Нет	Да	Да	Да
Представление иерархических данных	Нет	Да	Да	Да
Простота добавления новых знаний	Да	Нет	Нет	Да
Согласованность с ООП	Нет	Нет	Да	Да
Использование UML	Нет	Нет	Да	Да
Описание ИА	Нет	Нет	Нет	Да

Выбор фреймово-семантической модели представления знаний дает следующие преимущества:

- ✓ согласованность фреймово-семантической модели с концепцией объектно-ориентированного программирования обосновывает применение объектных языков программирования при разработке базы знаний и минимизирует затраты на создание программного обеспечения;
- ✓ единая модель представления знаний, которая подходит, как будет показано далее, и для предметной области мультиагентных

процессов преобразования ресурсов, и для предметной области проектирования информационных систем.

На основе выбранной модели представления знаний в параграфе 2.4.1 описывается концептуальная модель предметной области мультиагентных процессов преобразования ресурсов, а концептуальная модель предметной области проектирования информационных систем описана в параграфе 2.4.2.

2.4. Построение модели разработки информационной системы

2.4.1. Концептуальная модель предметной области мультиагентных процессов преобразования ресурсов

Основываясь на модели бизнес-процессов, выбранной в подглаве 2.3, можно построить следующую фреймово-семантическую модель мультиагентных процессов преобразования ресурсов (рис. 2.2) [16].

При автоматизации процессов предприятия любому преобразователю соответствует функция обработки данных, однонаправленная или двунаправленная (генерация, прием, передача, изменение, удаление).

Агент управляет объектами процесса принятия решений, выполняя следующие действия:

- ✓ анализирует внешние параметры (текущую ситуацию);
- ✓ диагностирует ситуацию, обращается к базе знаний, в случае определения соответствующей ситуации агент пытается найти решение (сценарий действий) в базе знаний или выработать его самостоятельно;
- ✓ принимает решение;
- ✓ определяет или переопределяет цели;
- ✓ контролирует достижение целей;
- ✓ делегирует цели своим и чужим объектам процесса преобразования ресурсов, а также другим агентам;
- ✓ обменивается сообщениями.

Архитектура агента мультиагентных процессов преобразования ресурсов основана на гибридной архитектуре InteRRaP и представлена на рис. 2.3.

Рис. 2.2. Семантическая сеть МППР

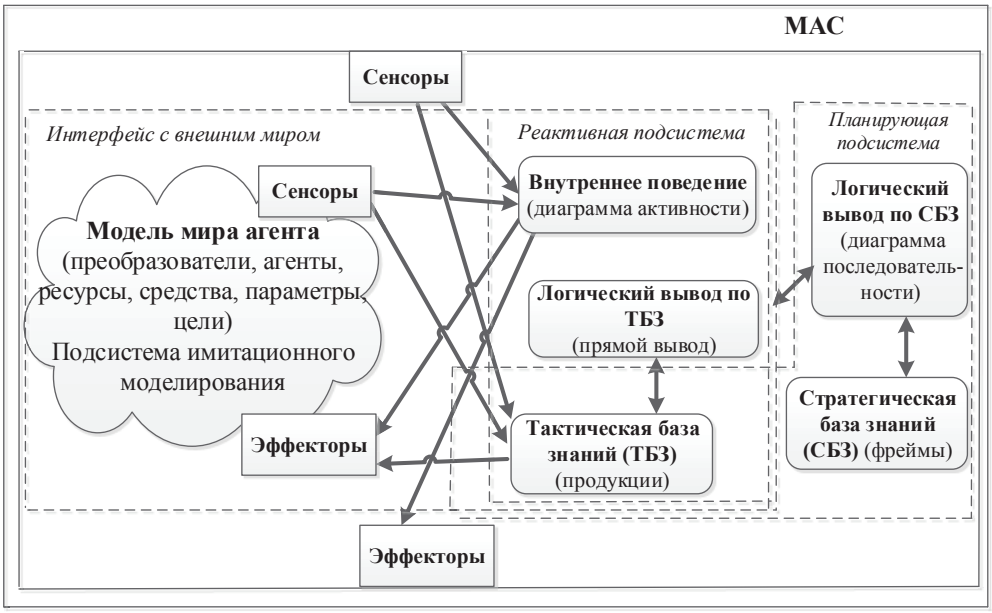

Рис. 2.3. Архитектура агента МППР

Модель организационно-технической системы, представленная в виде модели мультиагентных процессов преобразования ресурсов, показана на рис. 2.4.

Определим свойства и методы фрейм-концептов предметной области МППР. ФК перечисляются по мере их усложнения и согласно принципу «от общего к частному». В скобках у ФК указывается название родительского ФК, если такой есть.

ФК «Ресурс»

Свойства:

- ✓ идентификатор (*ID_RES*);
- ✓ название ресурса (*RES_Name*);
- ✓ тип ресурса (*kind*);
- ✓ текущее значение ресурса (RES_t);
- ✓ максимально возможное значение ресурса ($RES_{\max}$);
- ✓ начальное значение ресурса (RES_0);
- ✓ конечное значение ресурса (RES_T);
- ✓ цена единицы ресурса (*Cost*);
- ✓ время начала моделирования (t_0);
- ✓ единица измерения ресурса (*Metric_R*).

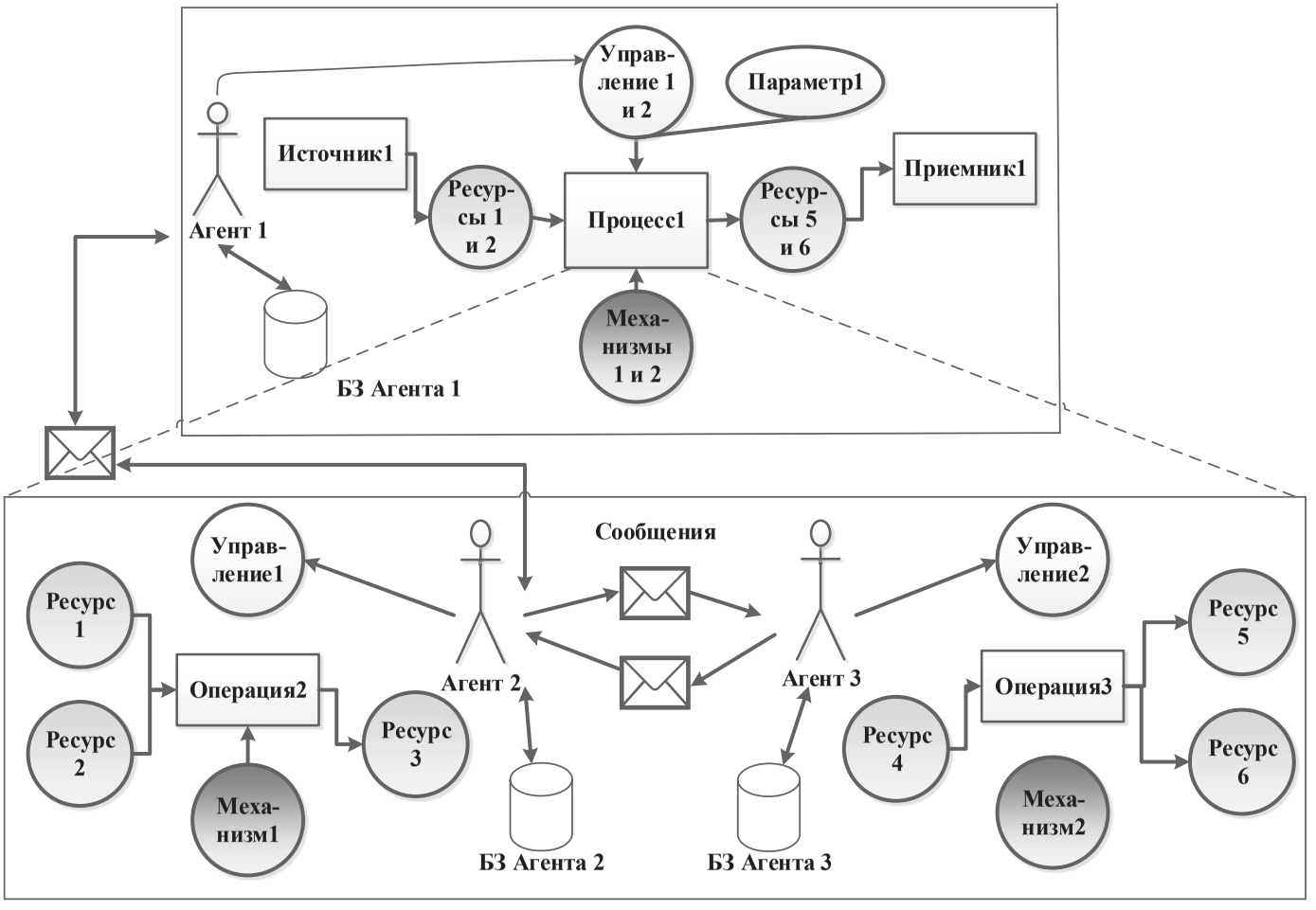

Рис. 2.4. Модель ОТС

Методы:
- ✓ суммарное приращение ресурса за интервал времени ($Hsum_{inc\_t}$);
- ✓ суммарное уменьшение ресурса за интервал времени ($Hsum_{dec\_t}$);
- ✓ стоимость ресурса (*Cost_sum*);
- ✓ приращение ресурса в текущий момент времени (H_{inc});
- ✓ уменьшение ресурса в текущий момент времени (H_{dec}).

ФК «Команда» (Ресурс)
Свойства:
- ✓ идентификатор (*ID_Cmd*);
- ✓ название ресурса (*Cmd_Name*);
- ✓ тип команды (*kind*);
- ✓ отправитель (*Msender*);
- ✓ получатель (*Mresiver*);
- ✓ текст сообщения (*text*);
- ✓ приоритет (*prior*);
- ✓ признак обработки (*read*);
- ✓ время создания (T_{create});
- ✓ время ожидания в очереди (T_{wait}).

ФК «Заявка» (Ресурс)
Свойства:
- ✓ идентификатор (*ID_Order*);
- ✓ название ресурса (*Order_Name*);
- ✓ заказываемый объем работ (*count*);
- ✓ выполненный объем работ (*real*);
- ✓ признак блокировки заявки (*lock*);
- ✓ имя элемента, обрабатывающего заявку (*Owner*);
- ✓ имя блока, создавшего заявку (*Parent*);
- ✓ приоритет заявки (*prior*);
- ✓ время создания заявки (T_{create});
- ✓ время ожидания заявки в очереди (T_{wait}).

ФК «Средство» (Ресурс, Операция)
Свойства:
- ✓ идентификатор (*ID_Mech*);
- ✓ название ресурса (*Mech_Name*);
- ✓ тип команды (*kind*);
- ✓ текущее количество свободных средств ($Mech_t$);

- ✓ всего средств ($Mech_{all}$);
- ✓ время создания (T_{create});
- ✓ состояние средства (*Status*);
- ✓ единоразовые затраты ресурсов при начале преобразования (RES_{in});
- ✓ единоразовые затраты ресурсов при окончании преобразования (RES_{out});
- ✓ расход ресурсов в единицу времени (RES_{use});
- ✓ единоразовые затраты ресурсов при захвате средства другой операцией (RES_{lock});
- ✓ единоразовые затраты ресурсов при освобождении (RES_{unlock});
- ✓ затраты ресурсов при возникновении и устранении поломки (RES_{other});
- ✓ периодичность возникновения поломки (T_{other});
- ✓ начальная цена единицы средства (*Cost*);
- ✓ суммарное время использования средства (T_{mech_use});
- ✓ суммарное время простоя средства (T_{mech_stand}).

Методы:

- ✓ действие по запуску средства в момент начала преобразования (Am_{in});
- ✓ действие по остановке средства в момент окончания преобразования (Am_{out});
- ✓ действие по выполнению преобразования (Am_{use});
- ✓ действие по остановке средства в момент прерывания преобразования (Am_{lock});
- ✓ действие по запуску средства в момент продолжения преобразования (Am_{unlock});
- ✓ действие по устранению поломки (Am_{other});
- ✓ производительность средства в единицу времени (*product*).

Операция является наиболее общим элементом МППР, остальные (источники, приемники, перекрестки) содержат усеченный набор свойств и методов, поэтому приведем только описания ФК «Операция». У ФК «Источник» есть только множество выходов. У ФК «Приемник» есть только множество входов. У ФК «Перекресток» есть модель его поведения.

ФК «Операция» (Преобразователь)

Свойства:

- ✓ множество входов (*IN*);
- ✓ множество выходов (*OUT*);

- ✓ множество ресурсов, необходимых для прерывания операции (RES_{lock});
- ✓ множество ресурсов, необходимых для продолжения выполнения операции, остановленной в результате прерывания (RES_{unlock});
- ✓ цели операции (*GOp*);
- ✓ средства преобразования (*MECH*);
- ✓ состояние операции (*Status*);
- ✓ длительность выполнения преобразования (*time*);
- ✓ приоритет операции (*prior*);
- ✓ тип приоритета (*kind_prior*);
- ✓ признак запрета прерывания (*break_off*).

Методы:

- ✓ функция, реализуемая операцией (*f*);
- ✓ $C_a^{message}$– условие наличия необходимых входных сообщений;
- ✓ C_a^{order}– условие наличия необходимых входных заявок;
- ✓ C_a^{in}– условие наличия необходимых входных ресурсов;
- ✓ C_a^{out} — условие учета ограничений выхода;
- ✓ C_a^{mech} — условие готовности необходимых средств;
- ✓ C_a^{status} — условие готовности к исполнению;
- ✓ C_a^{time} — условие запуска по времени;
- ✓ $Action_{in}^{Message}$ — захват входного сообщения;
- ✓ $Action_{in}^{Order}$ — захват входных заявок;
- ✓ $Action_{in}^{RES}$ — захват входных ресурсов;
- ✓ $Action_{in}^{MECH}$ — захват средств;
- ✓ $Action_{out}^{Message}$ — формирование выходных сообщений;
- ✓ $Action_{out}^{Order}$ — формирование выходных заявок;
- ✓ $Action_{out}^{RES}$ — формирование выходных ресурсов;
- ✓ $Action_{out}^{MECH}$ — освобождение захваченных средств.

ФК «Агент»

Свойства:

- ✓ идентификатор (*ID_Agent*);
- ✓ название агента (*Agent_Name*);
- ✓ цели агента (*GAgent*);
- ✓ приоритет агента (*prior*);
- ✓ база знаний агента (*KBAg*);
- ✓ количество входящих сообщений (*Messin_count*);

- ✓ количество исходящих сообщений (*Messout_count*);
- ✓ сценарий поведения (*SPA*);
- ✓ множество управляемых объектов МППР (*Control_objects*);
- ✓ множество агентов «начальников» (*AU*);
- ✓ множество подчиненных агентов (*AD*).

Методы:

- ✓ анализ мира (*Analize*);
- ✓ диагностирование ситуаций (*Diagnost*);
- ✓ поиск решения (*Search*);
- ✓ обработка целей;
- ✓ контроль достижения целей;
- ✓ делегирование целей;
- ✓ обмен сообщениями;
- ✓ взаимодействие с БЗ и БД.

ФК «Параметр»

Свойства:

- ✓ идентификатор (*ID_Param*);
- ✓ название параметра (*Param_Name*);
- ✓ текущее значение параметра (P_t);
- ✓ начальное значение параметра (P_0);
- ✓ описание параметра (*desc*), плановое значение (*plan*).

Метод — функция вычисления параметра (*f*).

Таким образом, в данном подразделе описаны основные ФК КМПО предметной области МППР.

2.4.2. Концептуальная модель предметной области информационных систем

Предлагаемая КМПО ИС позволяет показать структуру ИС и взаимосвязи между всеми ее составляющими [19]. На первом уровне фреймово-семантической сети находятся узлы, соответствующие программному обеспечению и базе данных ИС. Поскольку при моделировании и разработке ИС используют функциональный и объектно-ориентированный подходы (см. параграф 1.1.2), то она также содержит ФК элементов моделирования архитектуры ПО, включая ФК следующих диаграмм: функциональных (IDEF0), потоков данных (DFD) (рис. 2.5)

и диаграмм языка UML (прецедентов, последовательностей и классов, рис. 2.6–2.8). Эти стандарты не включают в себя описание конкретных реализаций операций, тем не менее соответствующие ФК содержат методы, аналогичные методам ФК «Операция». Это позволяет сохранить данную информацию при переходе от модели МППР к модели ИС, а затем использовать ее при подготовке программных модулей.

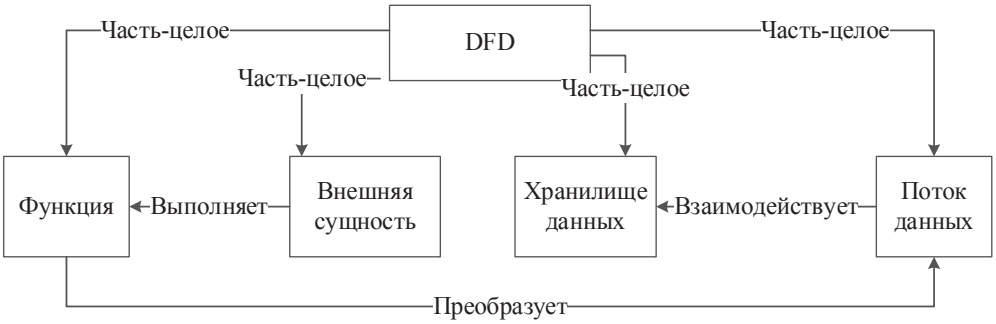

Рис. 2.5. Семантика DFD-диаграммы

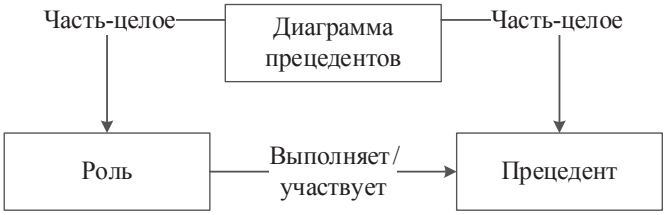

Рис. 2.6. Семантика диаграммы прецедентов

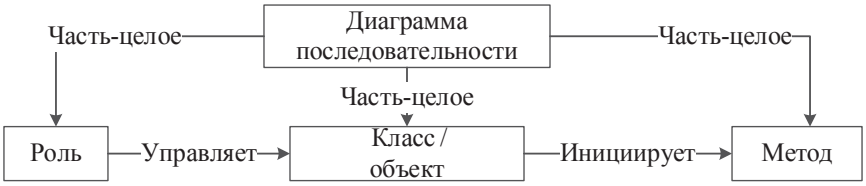

Рис. 2.7. Семантика диаграммы последовательности

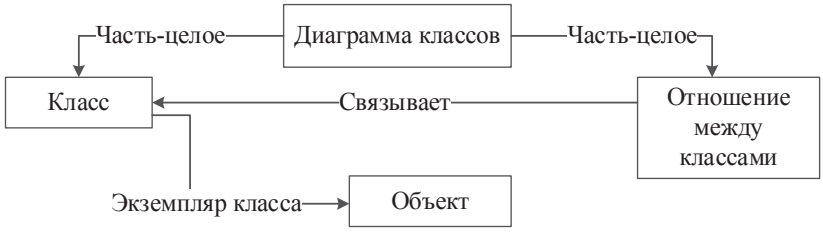

Рис. 2.8. Семантика диаграммы классов

Определим структуру фрейм-концептов.
1. Фрейм-концепты DFD-диаграммы

ФК «Функция»
Свойства:
- ✓ Идентификатор (*ID_F*);
- ✓ название (*F_Name*);
- ✓ множество входов (*IN*);
- ✓ множество выходов (*OUT*).

Метод — функция преобразования (*f*).

ФК «Внешняя сущность»
Свойства:
- ✓ идентификатор (*ID_Ex*);
- ✓ название внешней сущности (*Ex_Name*);
- ✓ цели внешней сущности (*ExG*);
- ✓ приоритет внешней сущности (*Exprior*).

ФК «Хранилище данных»
Свойства:
- ✓ идентификатор (*ID_DS*);
- ✓ название хранилища данных (*DS_Name*);
- ✓ структура хранилища данных (*DS_Structure*).

ФК «Поток данных»
Свойства:
- ✓ идентификатор (*ID_DF*);
- ✓ название потока данных (*DF_Name*);
- ✓ тип потока данных = {*Select, Insert, Delete, Update, Unknown*} (*DF_kind*);
- ✓ текущее значение потока данных (*DFt*);
- ✓ максимально возможное значение потока данных (*DFmax*);
- ✓ начальное значение потока данных (*DF0*);
- ✓ конечное значение потока данных (*DFT*);
- ✓ цена единицы потока данных (*DF_Cost*);
- ✓ единица измерения потока данных (*DF_Metric_R*).

2. Фрейм-концепты диаграммы прецедентов

ФК «Роль»
Свойства:
- ✓ идентификатор (*ID_R*);
- ✓ название роли (*R_Name*);
- ✓ цели роли (*RG*);
- ✓ приоритет роли (*Rprior*).

ФК «Прецедент»
Свойства:
- ✓ идентификатор (*ID_UC*);
- ✓ название (*UC_Name*).

Методы:
- ✓ функция преобразования (*UC_f*);
- ✓ условие запуска (*UC_Ca*).

3. Фрейм-концепты диаграммы классов

ФК «Класс»
Свойства:
- ✓ идентификатор (*ID_class*);
- ✓ название (*Class_Name*);
- ✓ другие свойства, определяются предметной областью.

Методы класса определяются предметной областью.

ФК «Объект»
Представляет собой экземпляр соответствующего класса, его свойства и методы определяются свойствами и методами соответствующего класса.

ФК «Отношение между классами»
Возможны следующие типы отношений [12]:
- ✓ ассоциации — произвольная взаимосвязь;
- ✓ обобщение — взаимосвязь между классом-родителем и классом-потомком;
- ✓ агрегация — взаимосвязь между классом-контейнером и классом-частью;
- ✓ композиция — агрегация, при которой с уничтожением класса-контейнера уничтожаются все его классы-части;

- ✓ зависимость — взаимосвязь между классами, при которой одному или нескольким классам требуются другие классы для их спецификации или реализации;
- ✓ реализация — взаимосвязь между классами, при которой один класс представляет некоторую спецификацию, а другой — его реализацию.

Свойства:

- ✓ идентификатор (*ID_rl*);
- ✓ название (*Rl_Name*);
- ✓ тип отношения (*Rl_type*);
- ✓ множество начальных классов (*Beg_classes*);
- ✓ множество конечных классов (*End_classes*).

Предлагаемый метод поддержки принятия решений в области моделирования и разработки информационных систем должен решать задачу перехода от концептуальной модели предметной области мультиагентных процессов преобразования ресурсов к концептуальной модели предметной области разработки информационных систем. Этот метод изложен в следующей подглаве.

2.5. Метод разработки информационных систем

Рассмотрим последовательность действий, позволяющих перейти от концептуальной модели предметной области мультиагентных процессов преобразования ресурсов к концептуальной модели предметной области разработки информационных систем (ФК DFD-диаграммы и ФК UML-диаграмм). Для описания семантики переходов между различными моделями БП и ИС используется диаграмма состояния объекта (стандарта IDEF5) [13].

Ресурс (*RES*)

Ресурс в ИС может представлять собой переменную:

$$RESin_i \rightarrow VAR_i,\ RESout_k \rightarrow VAR_k,$$

где $RESin_i$ — i-й входной ресурс; VAR_i — i-я переменная; $RESout_k$ — k-й выходной ресурс.

Для определения типа ресурса используется концептуальная модель предметной области разработки информационных систем. В ней отражены возможные типы данных для хранения ресурса.

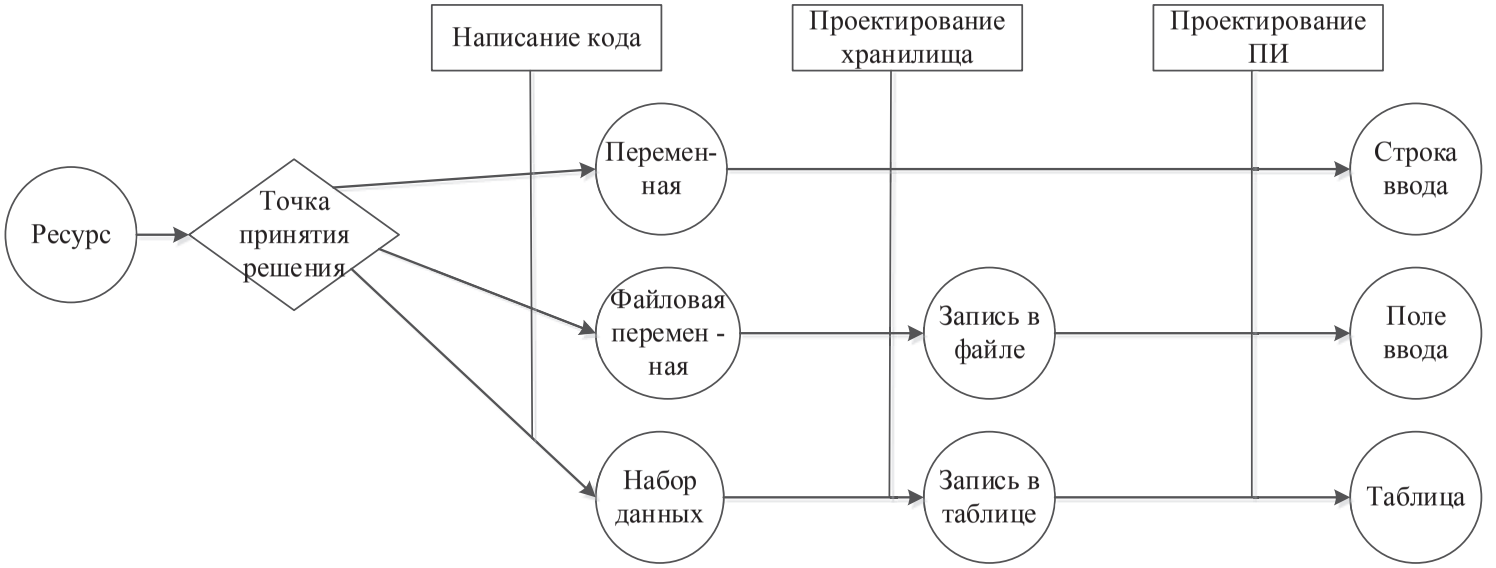

Рис. 2.9. Семантика перехода ресурса в элементы ИС

Возможны следующие варианты: простой тип, пользовательский тип, массив (на уровне переменных кода), таблица (на уровне базы данных). Семантика перехода ресурса в элементы информационной системы приведена на рис. 2.9. Семантика перехода ресурса в объекты диаграмм показана на рис. 2.10.

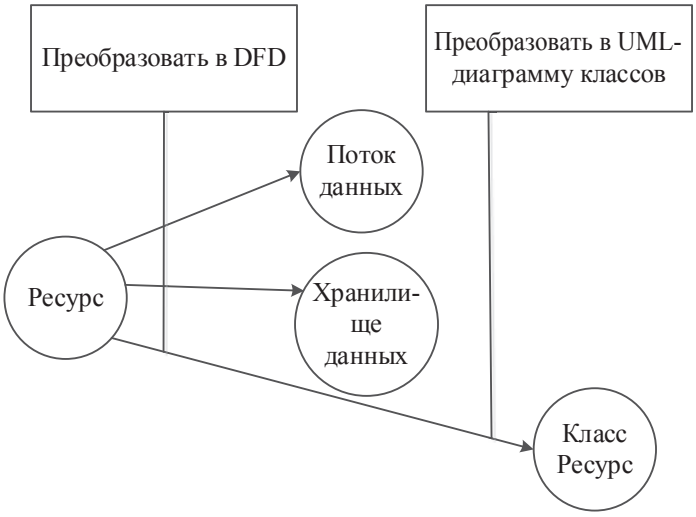

Рис. 2.10. Семантика перехода ресурса в объекты диаграммы

Сценарий перехода ресурса можно представить схемой (рис. 2.11).

Преобразователь (*Tr*)

К преобразователю относятся процессы, источники, приемники и перекрестки. Преобразователь как функция преобразования входных ресурсов в выходные может быть реализован в виде функции (ПО), которая выполняет обработку данных в памяти или на уровне файлов, или хранимой процедуры базы данных:

$Tr_i \rightarrow Func_i$, $Tr_i \rightarrow StoredProc_i$,

где Tr_i — преобразователь;

$Func_i$ — функция ПО;

$StoredProc_i$ — хранимая процедура.

При необходимости следует предусмотреть наличие элементов ПИ для определения условий начала преобразования. Входными параметрами для функции или ХП будут входные ресурсы, а выходными параметрами — выходные ресурсы. Таким образом, определив на 1-м этапе ресурсы, тем самым зададим типы параметров функций или ХП.

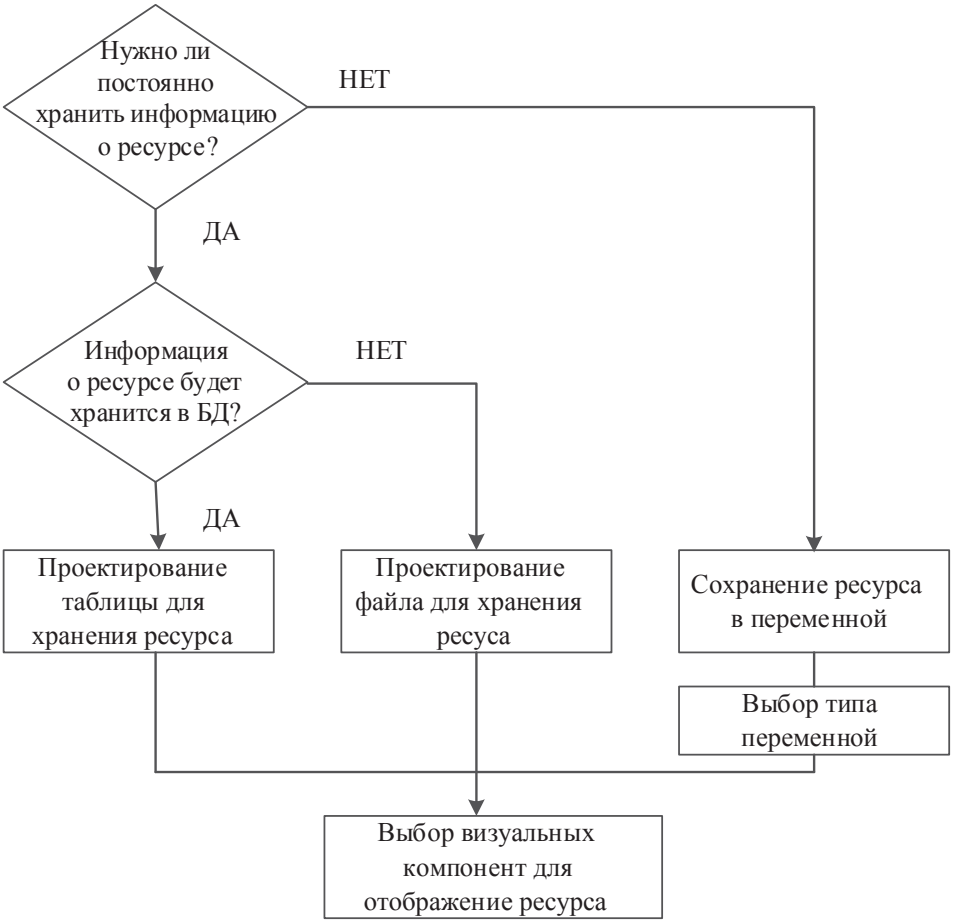

Рис. 2.11. Блок-схема перехода ресурса

В процессе интеллектуального моделирования и разработки ПО можно предоставить возможность описать алгоритм выполнения операции на алгоритмическом языке или T-SQL.

Семантика перехода преобразователя в объекты информационной системы приведена на рис. 2.12.

Семантика перехода ресурса в объекты диаграмм показана на рис. 2.13.

Переход от КМПО мультиагентных процессов преобразования ресурсов к КМПО информационной системы при работе с преобразователем определяется результатом перехода соответствующих ресурсов.

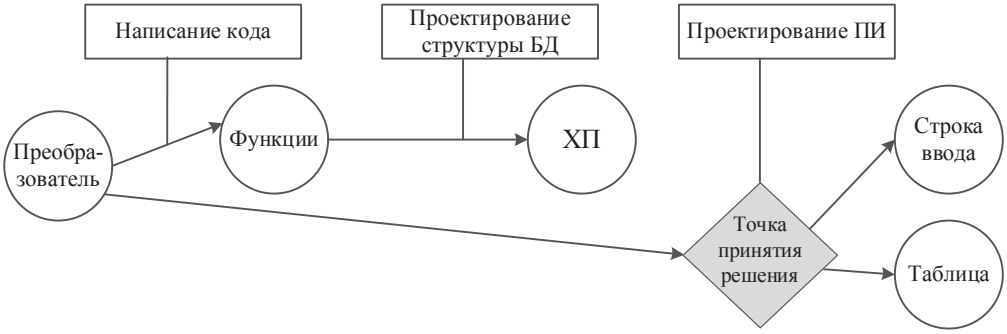

Рис. 2.12. Семантика перехода преобразователя в объекты ИС

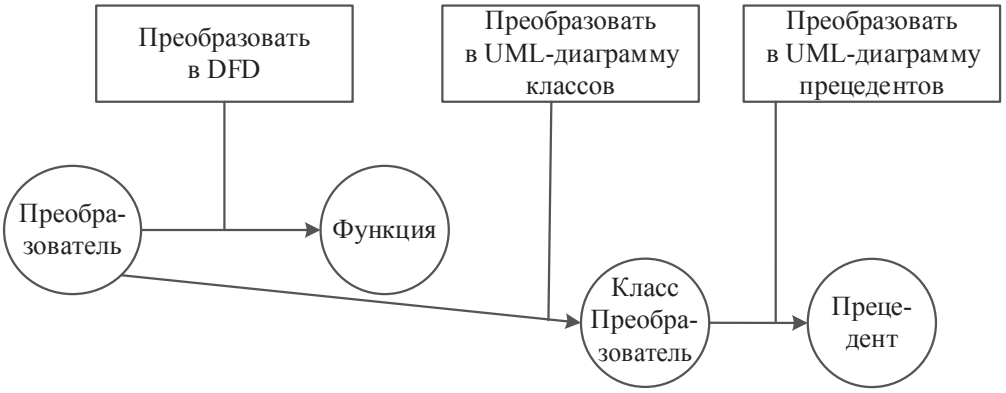

Рис. 2.13. Семантика перехода преобразователя в объекты диаграммы

Сценарий перехода преобразователя к элементам информационной системы показан на рис. 2.14. Написание алгоритма преобразования ресурса в предлагаемом методе возможно уже на уровне проектирования модели информационной системы.

Средства (*MECH)*

Средствами выполнения операций могут быть различные технические устройства, которыми оборудовано рабочее место человека, например контроллеры, датчики, компьютер, принтер, сканер. Информация об их характеристиках, полученная в процессе проектирования, идет в раздел «Технические требования к ПО». В некоторых ситуациях оператора этих устройств тоже можно рассматривать как средство. При моделировании и разработке ИС необходимо учитывать характеристики процессов обработки и передачи данных, так как в конечном итоге они оказывают влияние на характеристики основного процесса.

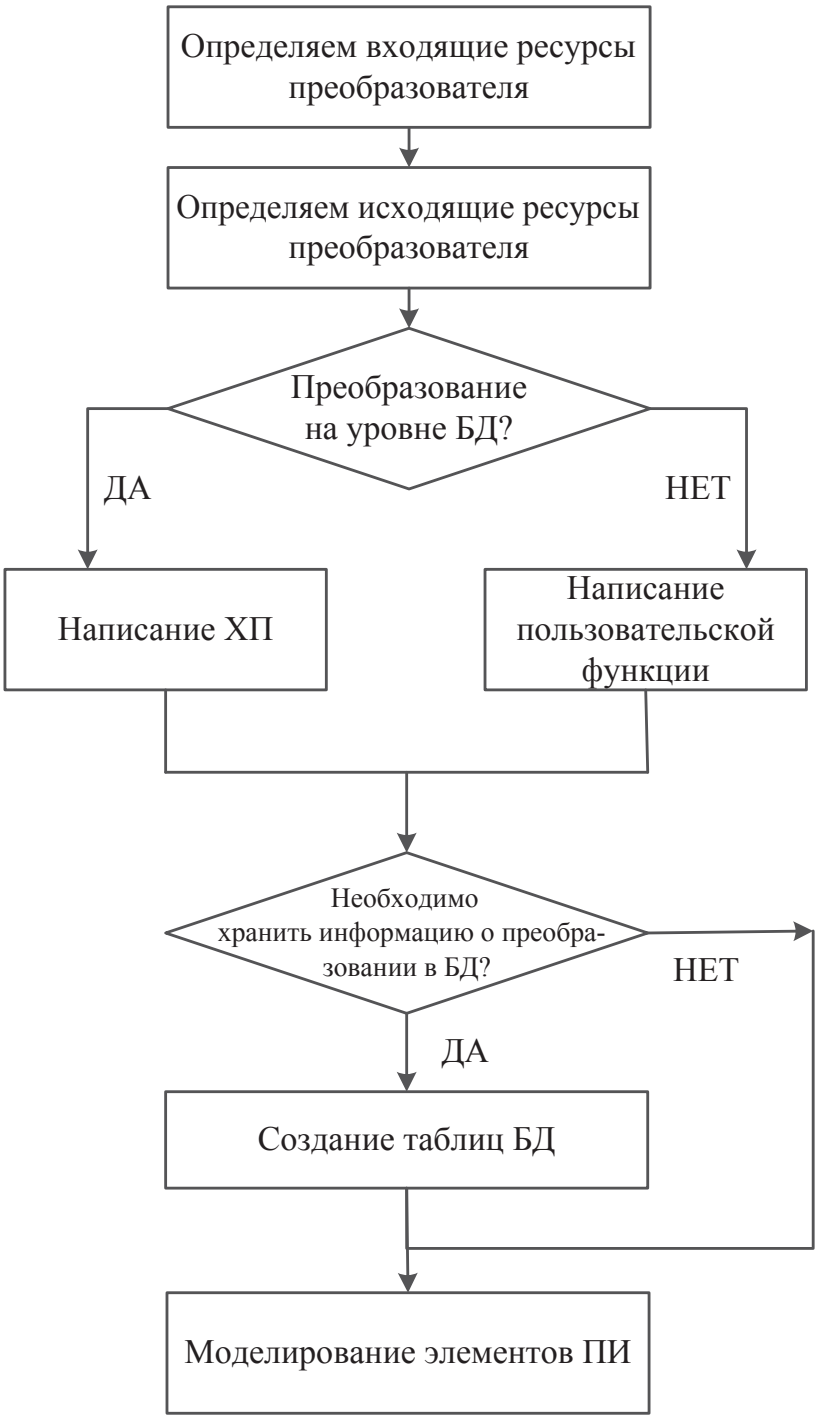

Рис. 2.14. Блок-схема перехода преобразователя

При разработке человеко-машинных программных комплексов при решении вопросов защиты информации средствам устанавливаются в соответствие определенные роли и назначаются права доступа к тем или иным объектам и функциям ИС. Таким образом, наблюдается аналогия с ресурсом. Поэтому за основу схемы проектирования средств взята схема проектирования ресурса.

Параметры (*P*)

Параметры (*P*) — это некоторая демонстрация пользователю характеристик протекания процесса или операции. В простейшем случае — это компонент «полоса прогресса», демонстрирующий как долго процесс еще будет работать. Также это может быть некоторое значение, вычисляемое по формуле, например какая-то характеристика ресурса. Следовательно, пользователь должен определить формулу и способ отображения параметра (график, текстовое поле):

$$P_k \rightarrow <Component_k, F_k>,$$

где P_k — k-й параметр; $Component_k$ — k-й визуальный компонент ПИ; F_k — k-я функция ПО.

Семантика перехода параметра представлена на рис. 2.15.

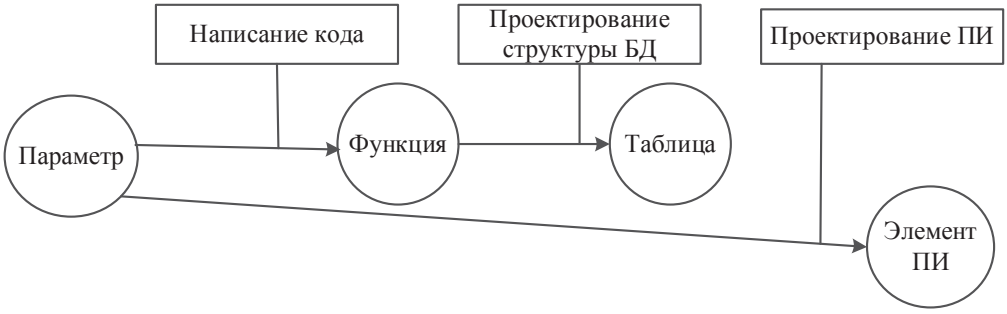

Рис. 2.15. Семантика перехода параметра

Сценарий перехода параметра показан на рис. 2.16.

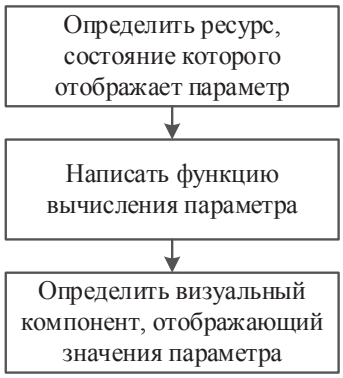

Рис. 2.16. Блок-схема перехода параметра

Агенты (*Agent*)

Агенту соответствует модель ЛПР, имеющая сложную структуру. При ее построении используют подходы искусственного интеллекта (например, в основе модели может лежать алгоритм или сценарий поведения, реализованный в виде ЭС; она включает в себя машину логического вывода, базу знаний, правила и рабочую память). С точки зрения ИС агент представляет собой программную сущность, у которой при необходимости имеется ПИ, функции, описывающие сценарий работы агента, и таблица для хранения БЗ агента. Возможна реализация сценария на уровне БД в виде ХП. Семантика перехода агента в элементы ИС приведена на рис. 2.17.

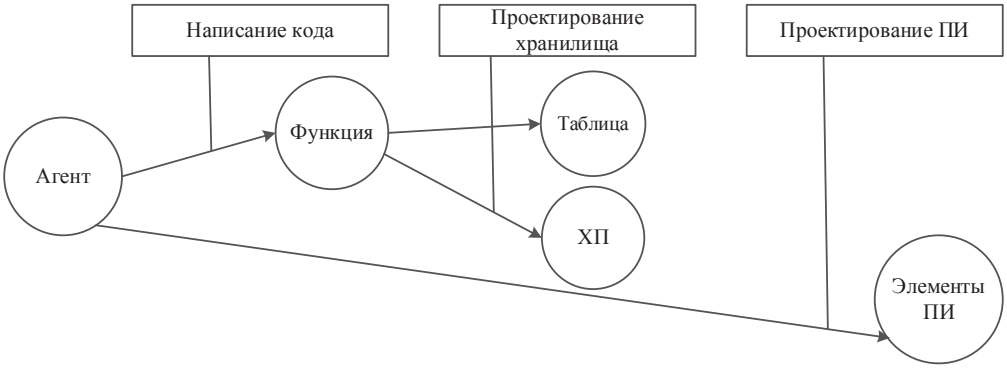

Рис. 2.17. Семантика перехода агента в элементы ИС

Семантика перехода агента в объекты диаграмм приведена на рис. 2.18. С помощью диаграммы последовательности моделируется сценарий поведения агента.

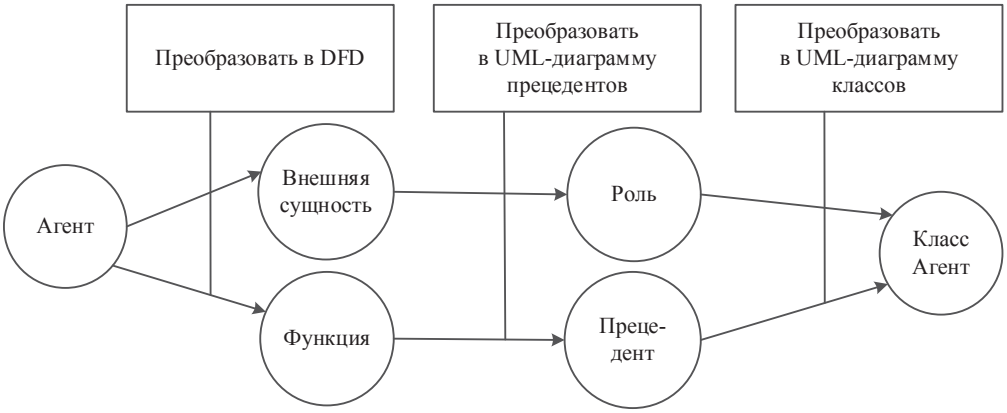

Рис. 2.18. Семантика перехода агента в объекты диаграмм

Рассмотрим сценарий перехода агента в объекты ИС (рис. 2.19).

Рис. 2.19. Блок-схема перехода агента

Далее приводится метод ППР разработки ИС предметной области МППР, состоящий из пяти следующих этапов [19]:

1. Разработка ИС начинается с обследования предметной области и построения имитационной модели МППР «Как есть».

2. На втором этапе проводятся имитационные эксперименты с моделью «Как есть» в целях выявления узких мест в организации процессов. По результатам моделирования строится модель МППР «Как будет».

3. На третьем этапе осуществляется построение модели ИС на основании данных из модели МППР.

Каждая операция из модели МППР, которую необходимо автоматизировать в ИС, преобразуется в функцию DFD-диаграммы. На диаграмме классов создается базовый класс операций, для каждой операции — экземпляр базового класса.

Все ресурсы, используемые в автоматизируемых операциях, преобразуются в потоки данных DFD-диаграммы, причем входные ресурсы *i*-й операции становятся входными потоками *i*-й функции DFD-диаграммы, а выходные ресурсы — выходными потоками *i*-й функции DFD-диаграммы. На диаграмме классов создается базовый класс ресурса, для каждого ресурса — экземпляр базового класса.

Для всех ресурсов модели необходимо создать хранилища данных на DFD-диаграмме. На UML-диаграмме классов создается базовый класс для хранилища данных.

Все агенты из модели МППР, которые будут реализованы программно, преобразуются во внешние сущности DFD-диаграммы. На диаграмме классов создается базовый класс агента, для каждого агента — экземпляр базового класса. На основании данных из DFD-диаграммы создаются диаграммы прецедентов. Каждая внешняя сущность преобразуется в актера соответствующей диаграммы прецедентов, а связанные с ней функции — в прецеденты.

Преобразование агента рассмотрим на примере агента с одним правилом («если» $a > b$, «то» $a = a - b$). Элементы памяти, необходимые для хранения переменных, преобразуются в хранилища данных DFD-диаграммы, правила «если» и «то» — в операции. В результате получаем DFD-диаграмму, представленную на рис. 2.20.

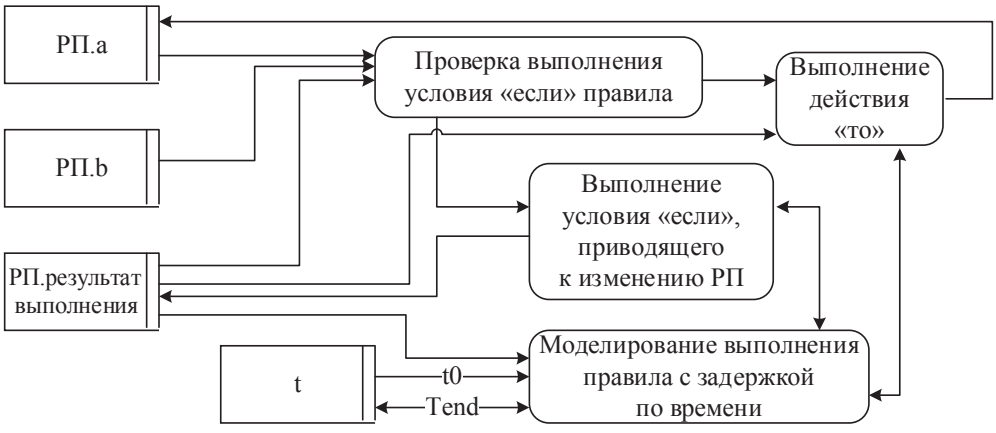

Рис. 2.20. Пример DFD-диаграммы для реактивного агента с одним правилом

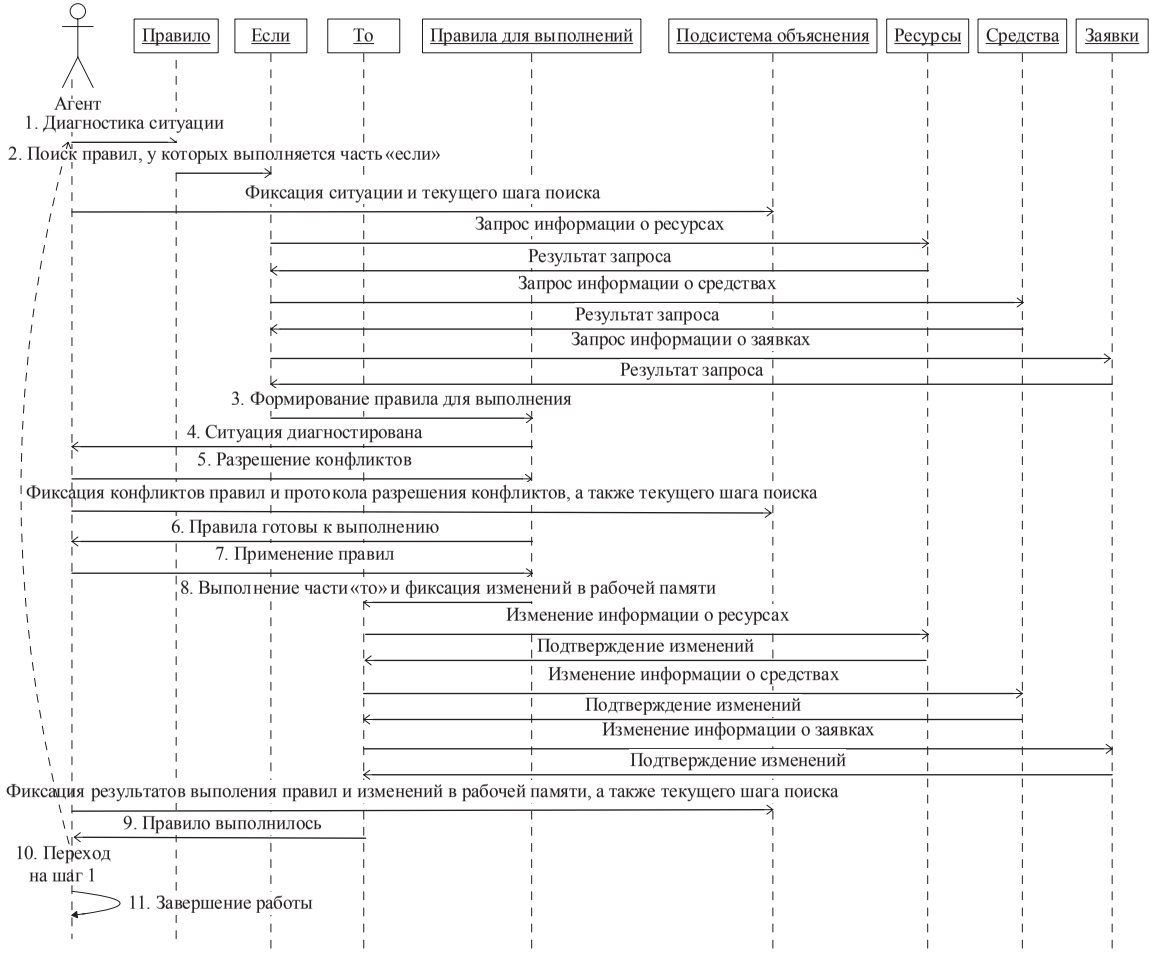

Рис. 2.21. Диаграмма поиска решения ИА проектирования ИС при конвертации агента на продукциях

Описание правил агента используется для построения диаграммы прецедентов, т. е. каждое правило переходит в прецедент.

Формулы, содержащиеся в условиях «если» и «то» правил агента переходят в описание метода соответствующего класса. На рис. 2.21 представлена диаграмма поиска решения ИА проектирования ИС при конвертации агента на продукциях. Отметим следующее:

- ✓ ресурсы, средства и заявки представляют собой рабочую память;
- ✓ если на шаге 2 ни одна ситуация не диагностирована, то происходит завершение работы.

База знаний ИА проектирования ИС представляет собой описание объектов МППР и ИС.

На основании данных из DFD-диаграммы создаются диаграммы прецедентов. Каждая внешняя сущность преобразуется в актера соответствующей диаграммы прецедентов, а связанные с ней функции — в прецеденты.

Атрибуты классов, соответствующих внешним сущностям, позволяют определить структуру таблиц ER-диаграммы.

4. На четвертом этапе система дорабатывается разработчиками, строится диаграмма последовательности и моделируется пользовательский интерфейс.

5. Для решения вопроса о размещении экземпляров концептов предметной области по базам знаний агентов рассмотрим стандартную задачу размещения с дискретным пространством решений, многократно описанную в литературе.

Найти $\min z = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij}$ при ограничениях $\sum_{j=1}^{n} a_{ij}x_{ij} \geqslant 1$, $i = 1, ..., m$, $x_{ij} = (0,1)$, $i = 1, ..., m; j = 1, ..., n$,

где m — количество экземпляров концептов предметной области; n — количество агентов; c_{ij} — коэффициент, показывающий величину затрат на размещение i-го экземпляра концепта у j-го агента; x_{ij} — коэффициент, показывающий принадлежность концепта агенту:

$$x_{ij} = \begin{cases} 1, \text{ если } i\text{-й экземпляр концепта} \\ \quad \text{расположен у } j\text{-го агента}, \\ 0, \text{ иначе}; \end{cases}$$

a_{ij} — коэффициент, определяющий потребность j-го агента в i-м концепте:

$$a_{ij} = \begin{cases} 1, \text{ если } i\text{-й экземпляр концепта} \\ \quad \text{нужен } j\text{-му агенту}, \\ 0, \text{ иначе.} \end{cases}$$

Таким образом, необходимо размесить экземпляры концептов по базам знаний агентов с минимальными затратами. Коэффициенты c_{ij} определяются экспертами и позволяют линейно упорядочить агентов по стоимости размещения экземпляров концептов с учетом стоимости разработки распределенной ИС и ее эксплуатации. В работе данная задача не рассматривается.

Постановка, приведенная для задачи размещения концептов по базам знаний агентов, имеет свою особенность. В имитационной модели БП имеется информация о том, какому агенту какие экземпляры концептов необходимы. Данная информация используется в ограничении задачи. По сути, известны допустимые варианты размещения концептов, среди которых нужно выбрать минимальный по затратам. Если потребности агентов в концептах достаточно обособлены друг от друга, то число возможных вариантов решения существенно сокращается по сравнению с полным перебором возможных вариантов.

Поскольку рассматриваемая задача решается в рамках автоматизации бизнес-процессов, то в сравнении с классической задачей необходимо учитывать, что действия агентов являются составной частью автоматизируемого бизнес-процесса и должны укладываться в определенные временные рамки $T_{\text{БП}}^{\max}$ — максимальное время выполнения бизнес-процесса. Следовательно, к ограничениям необходимо добавить ограничение на время выполнения бизнес-процесса ($T_{\text{БП}}$), которое в общем случае будет иметь вид

$$T_{\text{БП}} \leqslant T_{\text{БП}}^{\max}.$$

Рассмотрим решение этой задачи для случая разработки системы ведения реестров. Пусть необходимо разместить 5 (пять) реестров между 3 (тремя) агентами. Законодательно существует ограничение на выполнение операции — 3 дня. В случае нарушения сроков могут быть предъявлены существенные штрафы. Особенность данной системы состоит в том, что не все агенты равноправны. Один из них, центральный офис, должен хранить у себя все 5 реестров, а остальным двум агентам нужен доступ только к тем реестрам, с которыми они работают. Экспертные оценки затрат на размещение концептов предметной области по базам знаний приведены ниже.

Экспертные оценки затрат на размещение

Концепты	Агенты		
	1	2	3
1	2	1	1
2	2	1	1
3	2	1	1
4	2	1	1
5	2	1	1

Постановка задачи:

$$\min z = \sum_{i=1}^{5}\sum_{j=1}^{n} c_{ij} x_{ij}$$

при ограничении

$a_{11} + a_{21} + a_{31} + a_{41} + a_{51} \geqslant 1,$

$a_{12} + a_{22} \geqslant 1,$

$a_{23} + a_{33} + a_{43} \geqslant 1,$

$x_j = (0,1), j = 1, ..., 5,$

$T_{\text{БП}} \leqslant 3.$

Для наглядности представим ограничения в табличной форме: на пересечении i-й строки и j-го столбца стоит 1, если $a_{ij} = 1$.

Представление значений a_{ij} в виде таблицы

Агенты	Концепты				
	1	2	3	4	5
1	1	1	1	1	1
2	1	1	0	0	0
3	0	1	1	1	0

Рассмотрим возможные варианты решений.

1-й вариант. Размещение всех экземпляров концептов в центральном офисе. В таблице представлены значения x_{ij} и целевой функции z для 1-го варианта.

Решение для 1-го варианта

Агенты	Концепты					z
	1	2	3	4	5	
1	1	1	1	1	1	**10**
2	0	0	0	0	0	
3	0	0	0	0	0	

Все остальные варианты размещения приводят к дублированию концептов предметной области, поскольку по условию у центрального офиса обязательно должны быть все реестры. Для примера рассмотрим несколько вариантов размещения с дублированием.

2-й вариант. В таблице представлены значения x_{ij} и целевой функции z для 2-го варианта.

Решение для 2-го варианта

Агенты	Концепты					z
	1	2	3	4	5	
1	1	1	1	1	1	**14**
2	1	1	0	0	0	
3	0	0	1	1	0	

3-й вариант. В таблице представлены значения x_{ij} и целевой функции z для 3-го варианта.

Решение для 3-го варианта

Агенты	Концепты					z
	1	2	3	4	5	
1	1	1	1	1	1	**14**
2	1	0	0	0	0	
3	0	1	1	1	0	

4-й вариант. В таблице представлены значения x_{ij} и целевой функции z для 4-го варианта.

Решение для 4-го варианта

Агенты	Концепты					z
	1	2	3	4	5	
1	1	1	1	1	1	**12**
2	1	1	0	0	0	
3	0	0	0	0	0	

Современный уровень развития информационных технологий позволяет реализовать такую информационную систему с соблюдением временных ограничений на БП.

Поскольку специфика БП такова, что длительные простои в работе ИС могут привести к существенным штрафам, то необходимо оценить уровень надежности системы. Непосредственный расчет вероятностей

отказов и сбоев в сложной и многокомпонентной распределенной системе не представляется возможным.

Эксплуатационная надежность серверов и аппаратуры ИС оценивается обычно суммарными временами простоя в течение календарного года. Эти данные фиксируются в логах и могут быть легко проанализированы.

Для повышения надежности системы целесообразно использовать резервирование (резервирование замещением с ненагруженным резервом). Тогда в случае отказа основной системы подключается резервная, значительно более дешевая в эксплуатации и соответственно менее надежная. Очевидно, что система с ненагруженным резервом и надежным переключением дает повышение общей надежности на порядок против отсутствия резервирования.

На рис. 2.22 представлена схема БП в том случае, когда БЗ расположена в центре. В этом случае

$$T_{\text{БП}} = \text{t\_обработки}_{\text{A1}} + \text{t\_обработки}_{\text{Центр}} + \text{t\_передачи}_{\text{Центр-A1}},$$

где $\text{t\_обработки}_{\text{A1}}$ — время работы агента1, включая человеко-машинное участие; $\text{t\_обработки}_{\text{Центр}}$ — время работы Центра, включая человеко-машинное участие; $\text{t\_передачи}_{\text{Центр-A1}}$ — время передачи результатов работы из Центра агенту1.

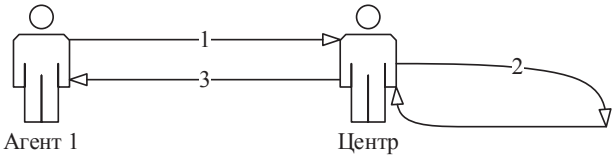

Рис. 2.22. Схема БП в случае расположения БЗ в Центре

Рассмотрим теперь возможные варианты резервирования системы.

1-й вариант. Размещение концептов, необходимых для работы агента у него (рис. 2.23). В этом случае

$$T_{\text{БП}} = \text{t\_обработки}_{\text{A1}} + \text{t\_передачи}_{\text{A1-Центр}} + \\ + \text{t\_обработки}_{\text{Центр}} + \text{t\_передачи}_{\text{Центр-A1}},$$

где $\text{t\_обработки}_{\text{A1}}$ — время работы агента1, включая человеко-машинное участие; $\text{t\_передачи}_{\text{A1-Центр}}$ — время передачи результатов работы из агента1 в Центр; $\text{t\_обработки}_{\text{Центр}}$ — время работы Центра, включая человеко-машинное участие; $\text{t\_передачи}_{\text{Центр-A1}}$ — время передачи результатов работы из Центра агенту1.

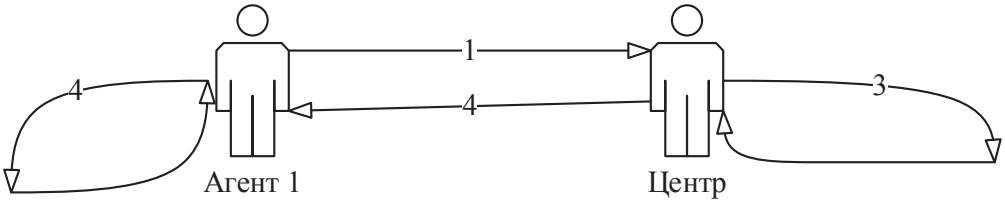

Рис. 2.23. Схема БП в случае размещения БЗ в Центре
и дублирования концептов у агентов

При отсутствии 2–3 дня возможности обмена информацией между агентом и Центром возможно $T_{\text{БП}} > T_{\text{БП}}^{\max}$. Следовательно, резервирование БЗ у агента может не решить проблемы.

2-й вариант. Размещение концептов, необходимых для работы агента у него, и резервный способ передачи сообщений в Центр (рис. 2.24).

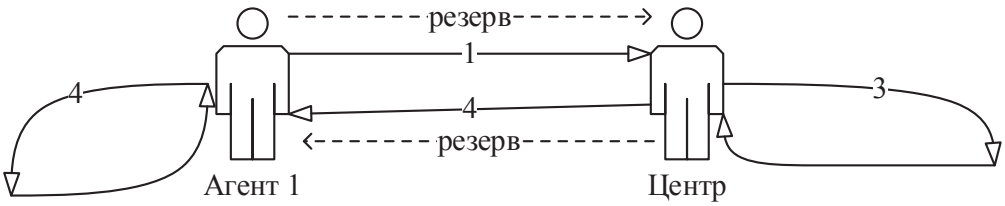

Рис. 2.24. Схема БП в случае распределенной ИС, дублирования концептов
у агентов и резервного способа передачи сообщений в Центр

В этом случае при сбоях в обмене информацией между Центром и агентом1 используется резервный способ доставки информации:

$$T_{\text{БП}} = \text{t\_обработки}_{\text{A1}} + \text{t\_передачи}_{\text{A1-Центр}} + \\ + \text{t\_обработки}_{\text{Центр}} + \text{t\_передачи}_{\text{Центр-A1}},$$

где $\text{t\_обработки}_{\text{A1}}$ — время работы агента1, включая человеко-машинное участие; $\text{t\_передачи\_рез}_{\text{A1-Центр}}$ — время передачи результатов работы из агента1 в Центр резервным способом; $\text{t\_обработки}_{\text{Центр}}$ — время работы Центра, включая человеко-машинное участие; $\text{t\_передачи\_рез}_{\text{Центр-A1}}$ — время передачи результатов работы из Центра агенту1 резервным способом.

Следовательно, для рассматриваемой задачи необходимо резервировать базу знаний агента и разрабатывать резервный способ передачи данных.

Анализ архитектурных решений человеко-машинной распределенной системы до начала процесса кодирования позволяет снизить риск отказа такой системы в процессе эксплуатации [20].

В табл. 2.3 приведено соответствие элементов МППР элементам ИС [19].

Таблица 2.3

Соответствие элементов МППР элементам ИС

Элементы МППР	Элементы DFD	Элементы UML диаграммы			Элементы ИС		
		прецедентов	классов	последовательности	ПИ	код	уровень БД
Ресурс	Поток данных, хранилище данных	Нет	Класс	Класс, параметр метода	Поле ввода, таблица	Переменная, файловая переменная, таблица	Таблица
Преобразователь	Функция	Прецедент	Метод класса	Метод	Строка ввода, таблица	Функция	ХП
Агент	Внешняя сущность	Актер	Класс	Актер, класс	Интерфейс программного модуля	Код программного модуля	Таблицы, ХП

Таким образом, предложенный метод позволяет преобразовать КМПО МППР в КМПО ИС.

Следует отметить, что на сегодняшний момент предлагаются методы разработки ИС так или иначе затрагивающие анализ процессов ОТС. Кратко рассмотрим их.

Метод П. О. Скобелева

Метод П. О. Скобелева предназначен для создания МАС оперативной обработки информации для поддержки процессов принятия решений. В качестве модели ОТС предлагается модель сети потребностей и возможностей (ПВ-сеть) предприятия [17]. При этом каждое предприятие представляется в виде сети агентов потребностей и возможностей. Метод решает задачу взаимодействия этих агентов в процессе принятия решений.

Метод разработки П. О. Скобелева включает в себя следующие этапы: описание предметной области МАС; описание классов аген-

тов и правил принятия решений; описание протоколов взаимодействия агентов; типов и структуры сообщений; программная реализация агентов.

Данный метод реализован в виде набора компонентов для разработки мультиагентных систем — MagentaToolkit [14]. Настройка системы под конкретную предметную область обеспечивается с помощью инструмента для создания онтологий. Инструментарий предназначен для разработки МАС, связанных с планированием и распределением ресурсов. Он не занимается вопросами анализа и реинжиниринга БП.

Метод О. В. Карсаева и В. И. Городецкого

Метод О. В. Карсаева и В. И. Городецкого базируется на методологии Gaia и среде MASDK [6], поддерживающей ее использование. Он предназначен для разработки прикладных многоагентных систем.

Метод состоит из десяти этапов, выполняемых в определенной последовательности [21]. Результаты каждого этапа определяют последовательность выполнения следующих этапов. Решения, описанные на одних этапах, являются исходными данными для выполнения последующих этапов. Укрупненно метод может быть описан в виде пяти стадий.

Первая стадия «Проектирование прикладной МАС» включает в себя два этапа:

- ✓ анализ предметной области;
- ✓ описание онтологии предметной области.

Результатом этапа анализа предметной области является задача идентификации классов агентов и сопоставления им ролей. Распределение ролей между классами агентов определяет, какие классы агентов на дальнейших этапах разработки будут обеспечивать решение определенных задач.

Вторая стадия «Проектирование классов агентов» занимается описанием трех компонент, образующих структуру агента:

- ✓ модель поведения агента;
- ✓ модели сервисов;
- ✓ ментальная модель.

На этой стадии идет только описание агентов с помощью диаграмм с использованием объектно-ориентированного подхода.

Написание программного кода происходит на третьей стадии. При этом описываются сервисные функции. После этого происходит автоматическая генерация программного кода классов агентов.

На четвертой стадии идет описание агента, состоящее из данных, необходимых для их установки, а также знаний и правил его поведения.

На последней стадии происходит развертывание агентов в сети.

Таким образом, метод О. В. Карсаева и В. И. Городецкого не позволяет описать статические и динамические БП, а следовательно, не занимается вопросами их анализа и реинжиниринга.

Метод А. Н. Швецова

Предложенный А. Н. Швецовым метод относится к разработке корпоративных интеллектуальных систем поддержки принятия решений [23]. На первоначальном этапе большое внимание уделяется описанию структурного, логического и поведенческого аспектов функционирования ОТС. На этапе формализации строятся структурно-логическая модель, база знаний, топологическая и объектная модели. Далее разрабатывается прототип системы и ее промышленный вариант. В данном методе основной упор делается на извлечение и формализацию знаний о предметной области.

Данный метод реализован в виде программного пакета DISIT (Distributed Intellectual System Integrated Toolkit). Он предназначен для разработки МАС, основан на следующих принципах:

- ✓ описание модели предметной области с использованием фрейм-концептов;
- ✓ описание поведения агентов в виде продукций.

В данном инструментальном пакете последовательно выполняются следующие этапы метода:

- ✓ представление модели предметной области;
- ✓ наполнение модели логикой взаимоотношений фрейм-концептов и их атрибутов;
- ✓ выделение интеллектуальных агентов и определение их поведения с учетом системных ограничений;
- ✓ трансляция полученной концептуальной модели предметной области в структурно-логическую модель МАС;
- ✓ размещение интеллектуальных компонент и агентов в корпоративной сети.

Таким образом, метод А. Н. Швецова не позволяет описать статические и динамические БП, а следовательно, не занимается вопросами их анализа и реинжиниринга.

Метод Д. В. Александрова

Александровым Д. В. предложен метод моделирования распределенных систем управления БП предприятия [10]. При анализе предметной области предлагается использовать структурно-функциональный подход. Имитационные модели разработаны на аппарате раскрашенных сетей Петри. По результатам имитационного моделирования предлагаются рекомендации по совершенствованию БП, а при необходимости проведение тактического реинжиниринга, заключающегося в добавлении (удалении) функций, сотрудников, в перераспределении функций между сотрудниками, переводе сотрудников из одного структурного подразделения в другое и т. п. Затем идет реализация агентного приложения для автоматизации выполнения БП. Также метод предполагает использование ИМ для мониторинга БП предприятия.

Для сравнительной оценки методов разработки информационных систем (табл. 2.4) предлагается следующий набор критериев:

- ✓ модель процессов организационно-технической системы, которая должна описывать статические и динамические бизнес-процессы, а также модель лица, принимающего решение;
- ✓ средства анализа процессов, включающие в себя организационный реинжиниринг, анализ «узких мест»;
- ✓ возможность использования данных из модели организационно-технической системы при разработке ИС;
- ✓ использование структурного и объектно-ориентированного подходов;
- ✓ результаты автоматизации (бизнес-процессы, согласование решений, процесс принятия решений — использование МЛВ).

Таблица 2.4

Сравнение методов разработки ИС

Критерии сравнения	Методы				
	Скобелева	Городецкого, Карсаева	Швецова	Александрова	Новый
1. Модель процесса ОТС:	ПВ-сети	Нет	Нет	Раскрашенные сети Петри	Модель МППР
статический бизнес-процесс	Да	Нет	Нет	Да	Да
динамический бизнес- процесс	Нет	Нет	Нет	Нет	Да
модель ЛПР	Да	Да	Да	Нет	Да

Окончание табл. 2.4

Критерии сравнения	Методы				
	Скобелева	Городецкого, Карсаева	Швецова	Александрова	Новый
2. Средства анализа процессов: реинжиниринг организационный	Нет	Нет	Нет	Да	Нет
анализ «узких мест»	Нет	Нет	Нет	Нет	Да
3. Разработка ИС: на основе модели ОТС в части динамических БП	Нет	Нет	Нет	Нет	Да
на основе модели ОТС в части ЛПР	Да	Да	Да	Нет	Да
структурный подход	Нет	Нет	Нет	Да	Да
ОО подход	Да	Да	Да	Нет	Да
4. Результаты автоматизации: бизнес-процессы	Да	Да	Да	Да	Да
согласование решений	Да	Нет	Да	Нет	Да
ППР — использование МЛВ	Нет	Нет	Да	Нет	Да

Существующие методы не полностью решают задачу разработки ИС, затрагивающую анализ процессов ОТС. Они не учитывают динамику БП, недостаточно уделяют внимание анализу «узких» мест, не используют информацию из модели процессов ОТС в части динамических БП для разработки ИС. Предлагаемый метод решает эти задачи. Кроме того, метод уделяет внимание вопросу надежности человеко-машинной распределенной системы в условиях ограничений по срокам выполнения бизнес-процессов.

2.6. Методика оценки эффективности работы метода разработки информационных систем

Оценим эффективность работы предложенного метода проектирования ПО. Одним из главных критериев оценки является время выполнения проекта по созданию ПО — $T_{\text{ПО}}$. Оно складывается из времени выполнения каждого этапа проекта и времени перехода между этапами. На практике на разных этапах разработки участвуют разные специалисты, поэтому требуется адаптация результатов предыдущего этапа для специалистов следующего этапа, поэтому и возникает время перехода между ними:

$$T_{\text{ПО}} = T_{\text{обсл}} + T_{\text{форм}} + T_{\text{модел}} + T_{\text{проект}} + T_{\text{разр}} + T_{\text{переход}},$$

где $T_{\text{обсл}}$ — время проведения обследования (1-й этап); $T_{\text{форм}}$ — время формализации БП, т. е. время построения модели; $T_{\text{модел}}$ — время проведения моделирования и реинжиниринга; $T_{\text{проект}}$ — время проектирования ПО; $T_{\text{разр}}$ — время разработки ПО; $T_{\text{переход}}$ — суммарное время, затрачиваемое при переходе от одного этапа проекта к другому:

$$T^{i\to j}_{\text{переход}} = T^{\text{обсл}\to\text{форм}}_{\text{переход}} + T^{\text{форм}\to\text{модел}}_{\text{переход}} + T^{\text{модел}\to\text{проект}}_{\text{переход}} + T^{\text{проект}\to\text{разр}}_{\text{переход}},$$

где $T^{i\to j}_{\text{переход}}$ — время перехода между i-м и j-м этапами.

Время проведения обследования $T_{\text{обсл}}$ и время перехода между 1-м и 2-м этапами $T^{\text{обсл}\to\text{форм}}_{\text{переход}}$ не зависит от того, использовался или нет предлагаемый метод.

В процессе разработки ИС строятся DFD-диаграммы и диаграммы языка UML. Следовательно, время проектирования складывается из двух составляющих:

$$T_{\text{проект}} = T_{DFD} + T_{UML} + T^{DFD\to UML}_{\text{переход}},$$

где T_{DFD} — время построения всех DFD-диаграмм; T_{UML} — время построения модели архитектуры ПО в виде UML-диаграмм; $T^{DFD\to UML}_{\text{переход}}$ — время перехода от построения DFD-диаграмм к UML-диаграммам. (На практике связано с передачей знаний от аналитика БП к архитектору ПО.) Время построения одной DFD-диаграммы определяется количеством функциональных блоков и потоков данных. Следовательно,

$$T_{DFD} = \sum (nt_{\text{бл}} + mt_{\text{поток}}),$$

где n — количество функциональных блоков; $t_{\text{бл}}$ — время рисования одного блока; m — количество потоков данных; $t_{\text{поток}}$ — время рисования одного потока.

В предложенном методе используют следующие три вида UML-диаграмм: прецедентов, последовательности и классов. Значит,

$$T_{UML} = T_{UC} + T_{Int} + T_{Class},$$

где T_{UC} — время построения всех диаграмм прецедентов; T_{Int} — время построения всех диаграмм последовательности; T_{Class} — время построения всех диаграмм классов.

Время построения одной диаграммы прецедентов, в общем случае, зависит от количества вариантов использования ПО. При ее автоматической генерации на основе данных из DFD-диаграммы эта зависимость незначительна по сравнению с ручным моделированием. Следовательно, при использовании предложенного метода время построения одной диаграммы прецедентов можно считать величиной постоянной (t_{UC}), тогда $T_{UC} = \sum t_{UC}$.

При использовании предлагаемого метода $T_{\text{переход}}^{DFD \to UML}$ будет существенно меньше за счет процесса автоматизации моделирования. При использовании метода снижается человеческий фактор потери части информации при переходе от одной модели к другой.

В описанном методе предлагается автоматически генерировать заготовки программных модулей, описывающих классы и формы ПИ. Следовательно, время, затраченное на переход от этапа проектирования к этапу разработки ПО $T_{\text{переход}}^{\text{проект} \to \text{разр}}$, сокращается по сравнению с другими методами.

Анализ предложенного метода показывает, что он позволяет уменьшить следующие показатели: $T_{\text{переход}}^{DFD \to UML}$, T_{UC}, $T_{\text{переход}}^{\text{проект} \to \text{разр}}$.

Вывод по главе 2

Во второй главе было проведено исследование предметной области МППР и описаны основные объекты КМПО МППР, а также предметной области ИС и построена КМПО ИС. Разработаны модель и метод в области создания ИС, которые отличаются от существующих:

- ✓ методикой системного анализа и моделью для формализации МППР, при этом учитывается наличие ЛПР, которые могут быть представлены в виде ИА;
- ✓ ИМ для проверки модели «Как будет» на этапе реинжиниринга БП и оценки производительности ИС;
- ✓ интеллектуальностью разработки ИС, включающей функциональный и объектно-ориентированный анализ, моделирование ПИ, формирование исполняемого кода ИС.

Анализ надежности архитектурных решений при автоматизации процессов ОТС в условиях ограничений по срокам выполнения БП особенно эффективен для географически распределенной системы.

Предложенный метод ППР разработки ИС мультиагентных процессов преобразования ресурсов позволяет уменьшить время, затрачиваемое на переход от функционального моделирования к объектно-ориентированному, а также от проектирования к разработке ПО.

3. CASE-средство BPsim.SD

3.1. Функциональные возможности пакета BPsim.SD

BPsim.SD представляет собой CASE-средство автоматизации процесса проектирования DFD, IDEF0 диаграмм, UML-диаграмм прецедентов, последовательности и классов, а также пользовательского интерфейса разрабатываемой информационной системы [19].

В рамках процесса моделирования архитектуры информационной системы BPsim.SD предлагает пользователю следующие возможности:

- ✓ описание бизнес-процессов, автоматизируемых разрабатываемой ИС, с помощью диаграмм стандарта DFD. Диаграммы DFD декомпозируемы до любого уровня детализации. Автоматическое создание DFD-диаграмм на основе модели мультиагентных процессов преобразования ресурсов;
- ✓ описание функций, выполняемых пользователями системы в рамках автоматизируемых процессов, с помощью диаграмм прецедентов. Допускается построение диаграмм прецедентов с «нуля», т. е. создание новой диаграммы, или с помощью автоматизированного конвертирования из диаграммы DFD, в которой выбранным из списка существующих на диаграмме DFD внешним сущностям ставятся в соответствие актеры, а функциям — прецеденты. Полученные конвертированием диаграммы редактируемы. Допускается строить неограниченное количество диаграмм прецедентов для каждой диаграммы DFD;
- ✓ для каждого прецедента, выполняемого пользователем, может быть дано описание последовательности элементарных операций в системе (для этого предназначены диаграммы последова-

тельности, которые могут быть созданы для любой функции диаграммы прецедентов), частичная автоматизация этого процесса (разработчику предлагается перенести на диаграмму последовательности выбранных актеров с диаграммы прецедентов). Для каждого объекта диаграммы последовательности определяется один из четырех типов по принадлежности к определенному пакету (границы, актеры, управление, бизнес-объекты);

- ✓ создание диаграммы классов и сопоставление объектов диаграммы последовательности (кроме границ) с классами этой диаграммы;
- ✓ проектирование визуальных форм моделируемого программного обеспечения (границ диаграммы последовательности): размещение на форме компонентов, привязка к компонентам методов и свойств классов и сохранение моделей форм в форматах .dfm и .pas, передача в дальнейшем данных модели программисту для импорта в среду Delphi и дальнейшей проработки алгоритмов;
- ✓ формирование отчетов о созданном проекте с изображениями спроектированных диаграмм и форм, вывод отчетов на печать и экспорт в текстовый редактор Word;
- ✓ сохранение проекта на сервере и загрузка с сервера MS SQL Server для редактирования.

3.2. Описание CASE-средства BPsim.SD

3.2.1. Общая структура CASE-средства BPsim.SD

CASE-средство BPsim.SD (SD) состоит из трех подсистем: создание DFD-диаграмм, UML-диаграмм, моделирование пользовательского интерфейса. Структура BPsim.SD представлена на рис. 3.1. Модель МППР строится в BPsim.MAS — мультиагентной системе динамического моделирования ситуаций. Для преобразования модели мультиагентных процессов преобразования ресурсов в модель информационной системы используется диалоговая программа-помощник.

Более подробно подсистемы рассмотрены в следующих подглавах.

После ввода идентификационных данных пользователя и установления соединения с сервером открывается главная форма системы (рис. 3.2).

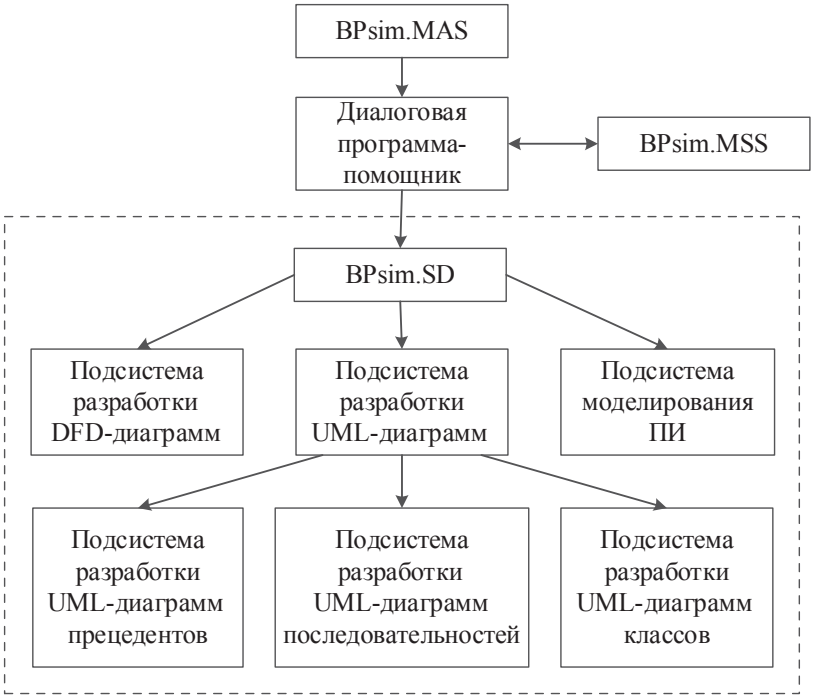

Рис. 3.1. Структура CASE-средства BPsim.SD

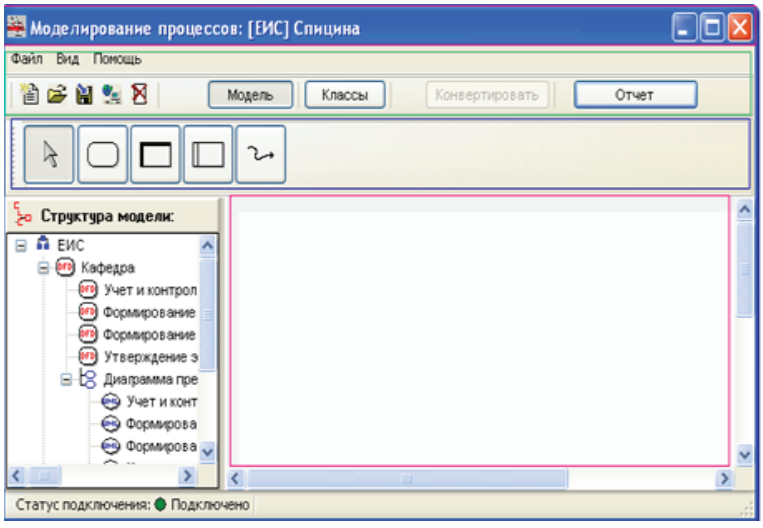

Рис. 3.2. Главная форма BPsim.SD

Главная форма состоит из следующих частей:

- ✓ заголовок, отражающий информацию о текущем проекте (название, авторы, дата и время создания);

- ✓ главное меню системы, предоставляющее пользователю доступ ко всем функциям приложения;
- ✓ панели инструментов, дублирующие основные команды главного меню (на них располагаются инструменты создания и редактирования диаграмм классов);
- ✓ область создания диаграммы, в которой происходит непосредственное создание и редактирование диаграмм классов;
- ✓ строка состояния, отображающая информацию о текущем состоянии подключения к выбранному серверу БД.

3.2.2. Создание диаграмм

После создания нового проекта в области панели инструментов автоматически активизируются кнопки с объектами диаграммы DFD (рис. 3.3).

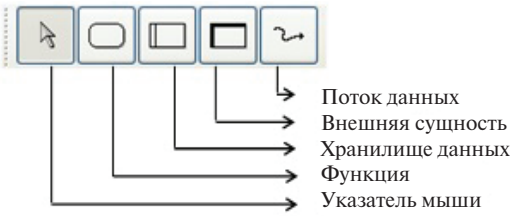

Рис. 3.3. Панель инструментов диаграммы DFD

Для размещения объекта в области диаграммы необходимо щелкнуть на соответствующей кнопке панели инструментов и в том месте диаграммы, куда требуется поместить объект.

Для соединения объектов потоками данных необходимо выбрать соответствующий объект в панели инструментов и навести курсор мыши на соединяемые объекты.

В контекстном меню **Тип** потока данных можно выбрать один из 5 типов (рис. 3.4).

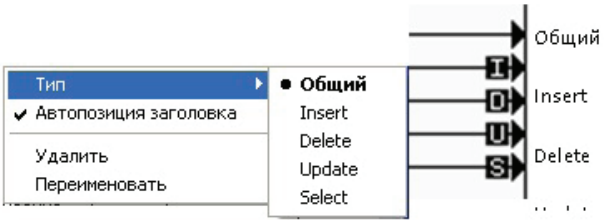

Рис. 3.4. Типы потоков данных

Все размещенные на диаграмме DFD функции заносятся в дерево проекта. Переименование объектов вызывается из контекстного меню объекта или соответствующего элемента дерева. Для потоков данных, хранилищ данных и внешних сущностей открывается дополнительное окно (рис. 3.5), позволяющее выбрать имя из списка уже существующих в данном проекте.

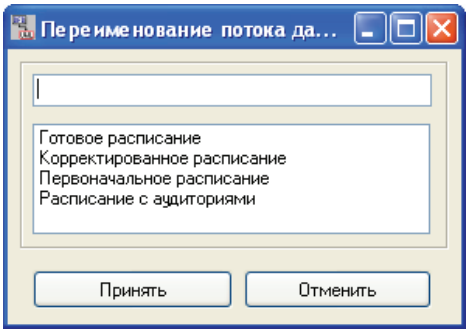

Рис. 3.5. Форма **Переименование потока данных**

Для создания декомпозиции диаграммы необходимо в контекстном меню декомпозируемой функции выбрать пункт **Декомпозиция**. Затем в открывшемся окне необходимо указать первоначальное количество элементов (функций) в декомпозиции (рис. 3.6).

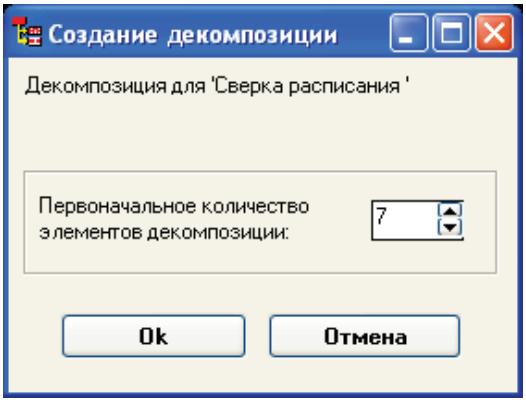

Рис. 3.6. Форма **Создание декомпозиции**

После нажатия на кнопку **OK** автоматически создается новая диаграмма, содержащая указанное количество функций. В дерево проектов эти функции помещаются в качестве потомков элемента — декомпозируемой функции.

Создать диаграмму прецедентов можно с помощью конвертирования из диаграммы DFD в соответствии с разработанной методикой. При этом функции диаграммы DFD заменяются на функции диаграммы прецедентов, а внешние сущности — на актеров.

Для запуска процесса конвертирования необходимо нажать кнопку **Конвертировать** на панели инструментов (рис. 3.7).

Рис. 3.7. Стандартная панель инструментов

В появившемся окне (рис. 3.8) выбираются существующие на данной диаграмме функции и внешние сущности, которые необходимо отобразить на диаграмме прецедентов. При необходимости сохранить связи между этими объектами устанавливается соответствующий флаг внизу окна.

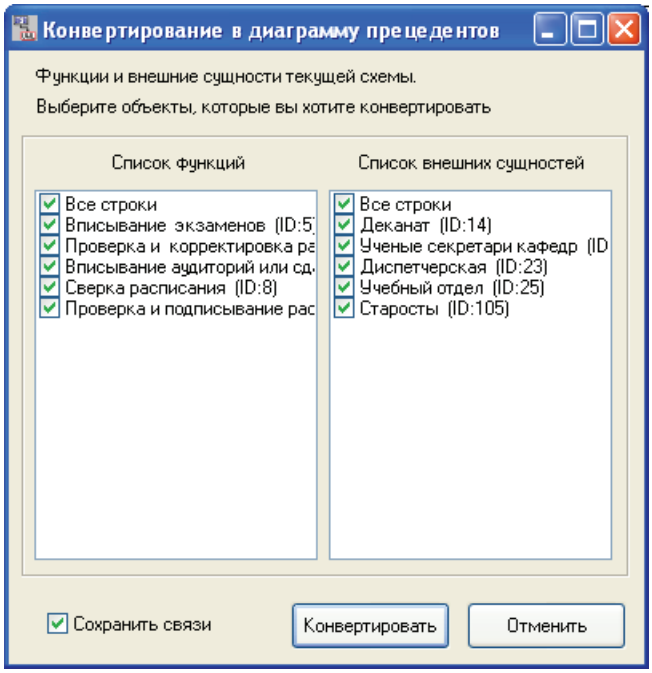

Рис. 3.8. Форма **Конвертирование в диаграмму прецедентов**

В результате конвертирования в дереве проектов появляется соответствующий узел со списком существующих на диаграмме прецедентов функций.

Разрешается создавать новую или редактировать существующую диаграмму с помощью панели инструментов диаграммы прецедентов (рис. 3.9).

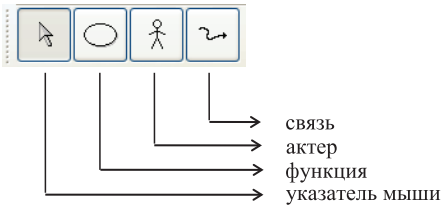

Рис. 3.9. Панель инструментов диаграммы прецедентов

Редактировать диаграмму прецедентов нужно аналогично редактированию диаграмм DFD.

Диаграмма последовательности может быть создана для любой функции диаграммы прецедентов из ее контекстного меню. В открывшемся окне (рис. 3.10) необходимо из списка выбрать существующих актеров, которые будут помещены на диаграмму последовательности.

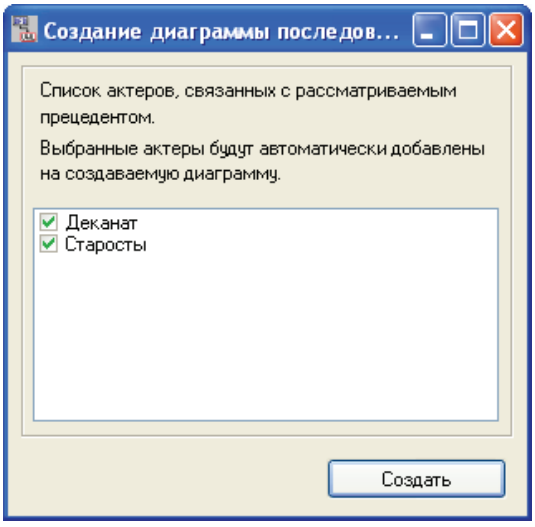

Рис. 3.10. Форма **Создание диаграммы последовательности**

После создания диаграммы соответствующий узел добавляется в дерево проектов и активизируется панель инструментов для редактирования данной диаграммы (рис. 3.11).

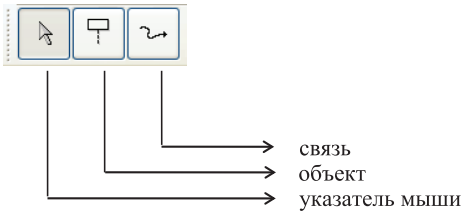

Рис. 3.11. Панель инструментов диаграммы последовательности

Редактирование диаграммы последовательности осуществляется аналогично редактированию диаграмм DFD.

Сохранение проекта на сервере осуществляется через пункт главного меню **Сохранить проект** или с помощью кнопки на стандартной панели инструментов (см. рис. 3.2).

Для перехода на диаграмму классов необходимо выбрать соответствующий пункт главного меню или воспользоваться панелью инструментов. Редактирование диаграммы классов выполняется с помощью панели инструментов, приведенной на рис. 3.12.

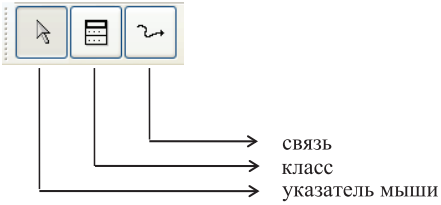

Рис. 3.12. Панель инструментов диаграммы классов

Для размещения объекта в области диаграммы необходимо щелкнуть на соответствующей кнопке панели инструментов и в том месте диаграммы, куда требуется поместить объект.

Двойной щелчок на созданном классе или выбор пункта **Свойства** контекстного меню открывают окно для ввода информации об этом классе (рис. 3.13). Окно содержит 3 вкладки:

- ✓ вкладка **Общие** (задаются имя и описание класса);
- ✓ вкладка **Свойства** (задаются свойства класса с указанием имени, типа данных и вида доступа);
- ✓ вкладка **Методы** (задаются методы класса с указанием имени, типа возвращаемого значения, вида доступа и входных параметров).

а *б* *в*

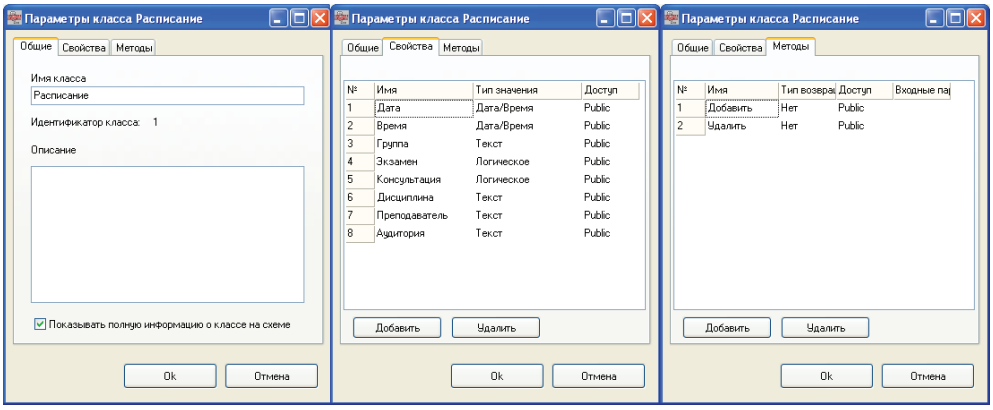

Рис. 3.13. Форма **Параметры класса**:
а — **Общие**; *б* — **Свойства**; *в* — **Методы**

3.2.3. Подсистема моделирования пользовательского интерфейса

Моделирование пользовательского интерфейса возможно при наличии в проекте диаграммы последовательностей и созданного на ней объекта-границы. В контекстном меню объекта-границы выбирается пункт **Форма**. У каждой границы может быть только одна форма, поэтому по нажатии на соответствующий пункт меню откроется или новая форма, или уже созданная.

Менеджер пользовательских форм вызывается из контекстного меню объекта **Границы диаграммы последовательности** (рис. 3.14).

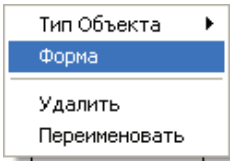

Рис. 3.14. Контекстное меню объекта
Границы диаграммы последовательности

Графический интерфейс менеджера форм приведен на рис. 3.15.

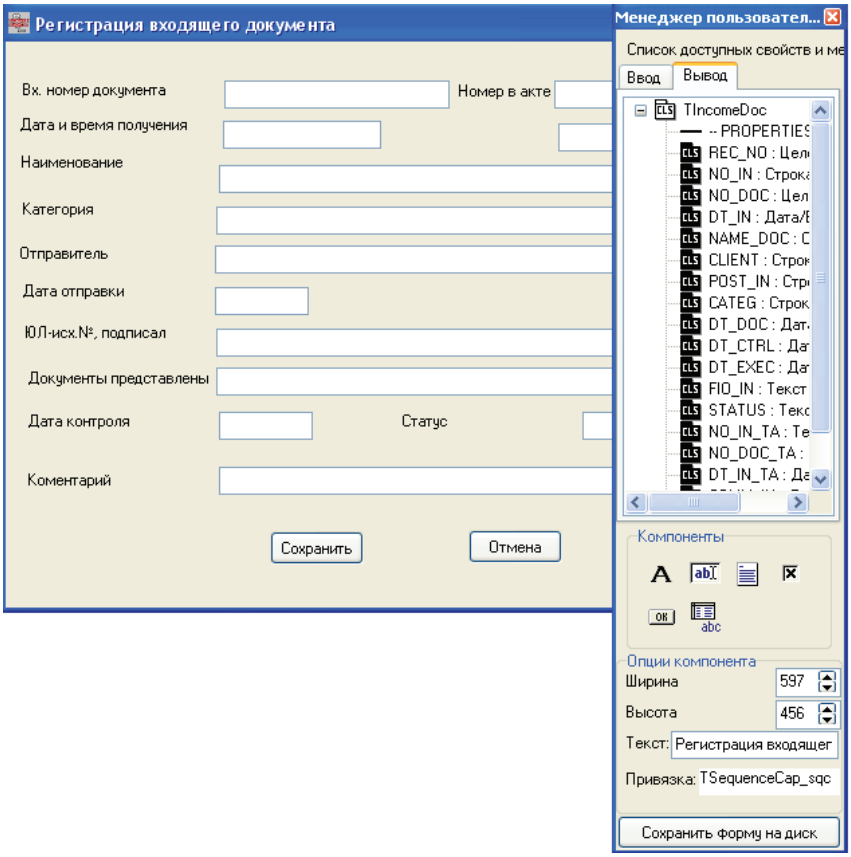

Рис. 3.15. Форма **Менеджер пользовательских форм** и прототип формы

Менеджер форм содержит следующие элементы:

- ✓ дерево входных и выходных классов проекта со списками доступных свойств и методов этих классов;
- ✓ панель визуальных компонентов, размещаемых пользователем на проектируемой форме;
- ✓ панель свойств компонентов, помещенных на форму.

Если ни один компонент не выбран, то на панели опций выводятся свойства проектируемой формы.

Для спроектированной формы файлы *.pas и *.dfm формируются с помощью кнопки **Сохранить форму на диск** и сохраняются на компьютере в указанном месте. В дальнейшем эти файлы могут быть открыты в Delphi для дальнейшей доработки.

Пример спроектированной формы **Расписание экзаменов** приведен на рис. 3.16.

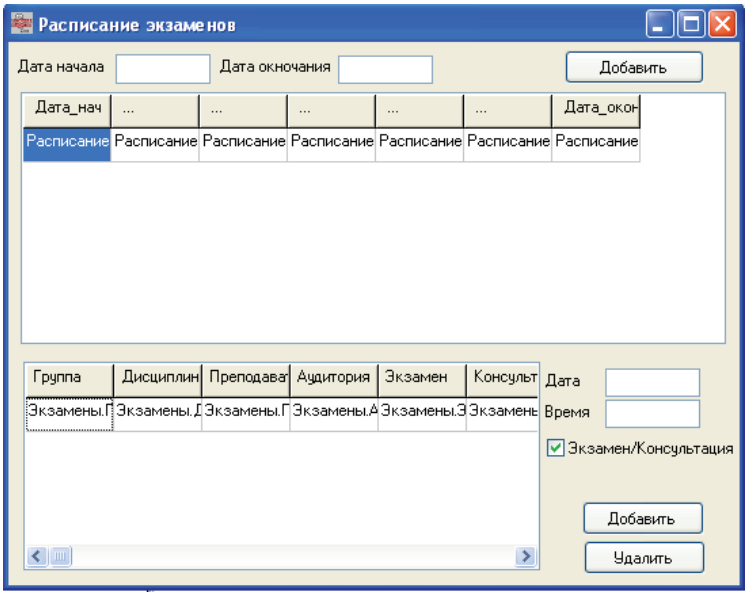

Рис. 3.16. Спроектированная в BPsim.SD форма составления расписания экзаменов

3.3. Описание агента интеграции BPsim.MAS и BPsim.SD

Агент интеграции BPsim.MAS и BPsim.SD доступен пользователям системы BPSim.MAS (меню **Общие/Агенты**). На начальном этапе появляется форма, позволяющая выбрать необходимую модель предметной области (рис. 3.17).

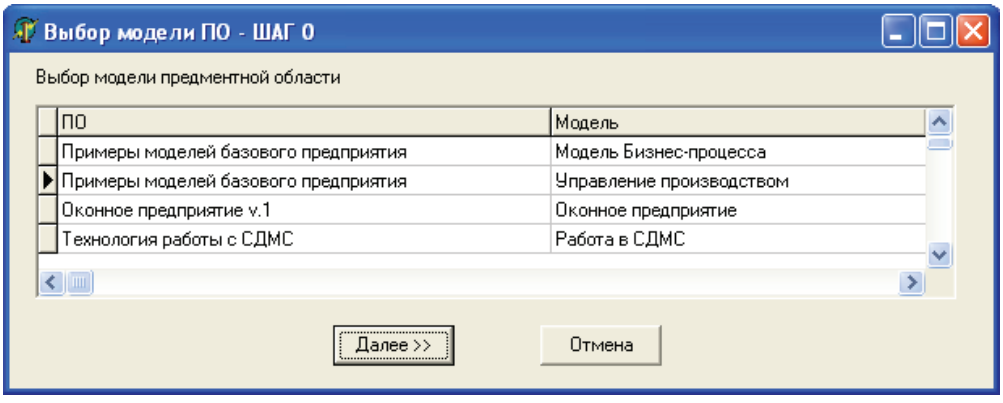

Рис. 3.17. Форма первого этапа конвертации моделей

После этого предлагается отобрать операции модели (рис. 3.18).

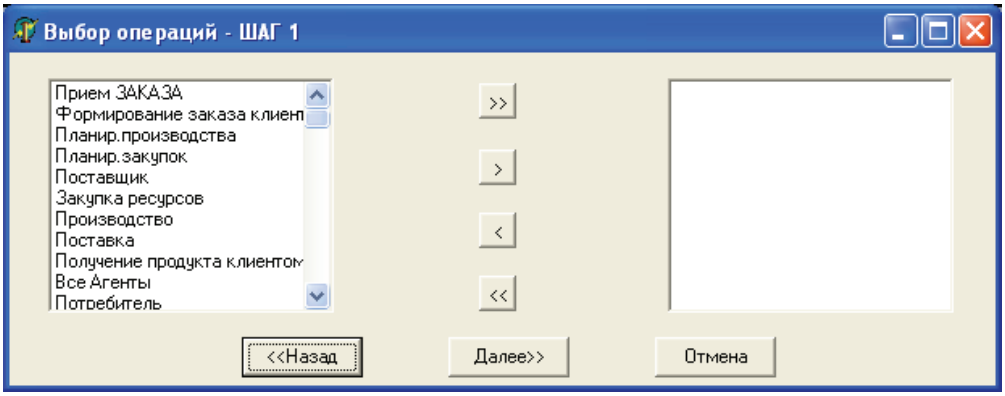

Рис. 3.18. Форма второго этапа конвертации моделей

На следующем этапе агент последовательно конвертирует элементы модели МППР в соответствующие элементы модели ИС (см. табл. 2.3).

3.4. Методика использования пакета BPsim

В процессе бизнес-моделирования и проектирования ПО предметной области МППР используются следующие пакеты программ (URL: http//www.bpsim.ru):

- ✓ BPsim.MAS (мультиагентная система динамического моделирования ситуаций);
- ✓ BPsim.MSS (интеллектуальная система технико-экономического проектирования);
- ✓ BPsim.SD (CASE-средство).

Методика бизнес-моделирования и проектирования ПО предметной области МППР в стандарте IDEF0 показана на рис. 3.19.

По результатам обследования БП аналитик с помощью BPsim.MAS создает модель БП предприятия «Как было», проводит моделирование и, если это необходимо, строит модель «Как будет». Результаты моделирования позволяют обосновать предложенные изменения в модели бизнеса. Результирующая модель «Как будет» конвертируется в DFD-диаграммы пакета BPsim.SD, учитываются только те процессы, которые будут автоматизированы. Этот пакет используется для проектирования ИС.

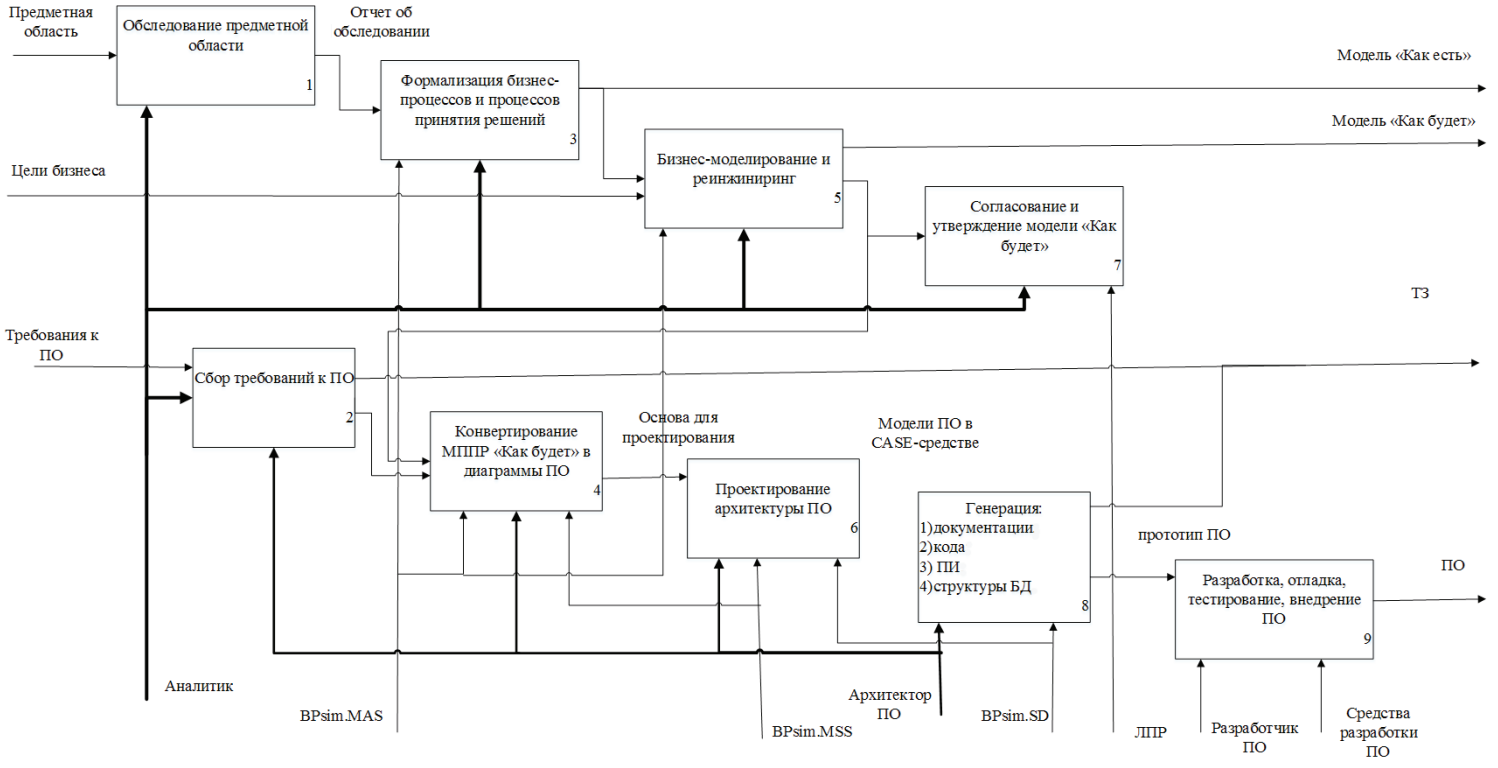

Рис. 3.19. Методика бизнес-моделирования и проектирования ПО

На основе данных DFD-диаграмм могут быть получены UML-диаграммы прецедентов, а также шаблоны классов и диаграмм последовательности, заготовка структуры БД. Затем архитектор ПО строит диаграммы последовательностей и классов, а разработчик ПИ — шаблоны форм. Данные заготовки будущей ИС могут быть преобразованы в программный код, который дорабатывается программистами.

Программная реализация интеграции BPsim.MAS и BPsim.SD возможна благодаря тому, что данные этих двух продуктов хранятся в единой БД. Конвертацию информации из одной системы в другую осуществляют диалоговые программные агенты-помощники.

Агенты-помощники, входящие в состав пакета BPsim, выполняют следующие функции:

- ✓ осуществляют передачу информации между программными продуктами в рамках комплексного решения единой задачи;
- ✓ облегчают работу непрограммирующего пользователя в процессе ознакомления с продуктами семейства BPsim;
- ✓ реализуют функцию проверки на этапах создания модели МППР, проектирования КМПО ИС.

DFD-диаграмма, отражающая последовательность действий агента-помощника при конвертации модели МППР в модель ИС, представлена на рис. 3.20.

Преимущество метода интеграции программных продуктов при помощи агентов-помощников заключается в возможности для непрограммирующего пользователя комплексно решать задачи моделирования бизнеса и разработки ИС. Кроме того, интеграция дает возможность существенно упростить и ускорить работу аналитиков и проектировщиков.

Применение метода и BPsim.SD для проектирования различных ИС представлено в [3, 15, 19].

Данная методика и продукты линейки BPsim позволяют комплексно решать задачи моделирования бизнеса, технико-экономического проектирования, моделирования архитектуры ИС и разработки прототипа приложения, что в итоге позволяет существенно упростить и ускорить работу аналитиков, а также повысить качество готового ПО за счет автоматизации некоторых этапов и сокращения влияния человеческого фактора.

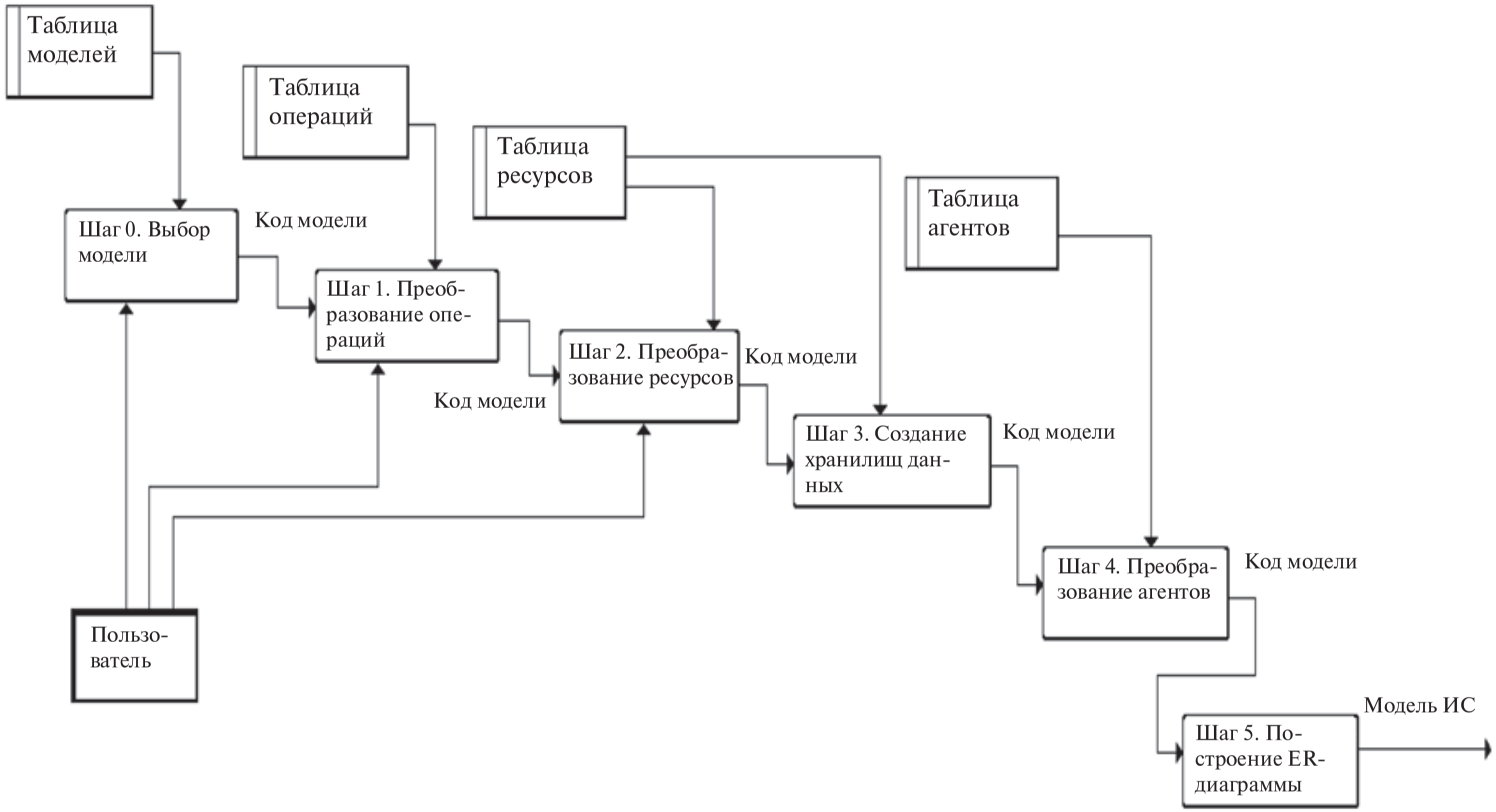

Рис. 3.20. DFD-диаграмма преобразования модели МППР в модель ИС

Выводы по главе 3

Решение задачи интеграции имитационного, экспертного, ситуационного и мультиагентного моделирования, а также функционального и объектно-ориентированного подхода позволило реализовать СППР в области разработки ИС BPsim.SD.

В третьей главе дано подробное описание возможностей BPsim.SD: функциональная и объектно-ориентированная разработка информационных систем, моделирование интерфейса пользователя. Приводится методика использования пакета BPsim и описания агента-помощника для преобразования модели МППР в модель ИС.

Заключение

1. Определен перечень характеристик и проведен сравнительный анализ наиболее распространенных CASE-средств: ARIS Toolset, Power Designer, Borland Together Designer, продукты IBM Rational, CA Erwin Modeling Suite, BizAgi и Elma. Обследованные системы имеют ряд недостатков: отсутствие интеллектуальности процесса проектирования — не решена задача автоматического перехода к проектированию одних диаграмм на основе других; не позволяют на основе динамической модели процессов организационно-технической системы (МППР) построить модель информационной системы.

2. Разработаны требования к моделям и методу поддержки принятия решений: динамическое моделирование МППР, содержащих модели интеллектуальных агентов, представляющих лиц, принимающих решение; имитационное моделирование для проверки модели «Как будет» на этапе реинжиниринга бизнес-процессов; интеллектуальное проектирование информационных систем, включающее функциональный и объектно-ориентированный анализ, проектирование интерфейса пользователя, формирование исполняемого кода информационной системы и структуры базы данных.

3. Решена задача системного анализа предметной области организационно-технических систем: выявлены три характерных класса процессов — бизнес-процессы, процессы согласования и принятия решений. Определены разрывы в модели знаний, возникающие между пользователем, аналитиком и разработчиком, приводящие к ошибкам при автоматизации организационно-технических систем.

4. В результате проведения анализа существующих моделей динамических бизнес-процессов и процессов принятия решений (сети Петри, расширенные сети Петри, системы массового обслуживания, мультиагентные процессы преобразования ресурсов) и знаний (про-

дукционная модель, семантическая сеть, фреймовая модель, фреймово-семантическая модель) были выбраны следующие модели. Для решения поставленной задачи в качестве модели организационно-технической системы предложено использовать динамическую модель мультиагентных процессов преобразования ресурсов К. А. Аксенова, поскольку она наиболее полно отвечает следующим требованиям: учет временных характеристик, возможность учета различных типов ресурсов, моделирование конфликтов на общих средствах, наличие модели интеллектуального агента (лица принимающего решение). Выбранная фреймово-семантическая модель представления знаний А. Н. Швецова имеет следующие преимущества: эффективно реализует иерархическое представление данных, хорошо сочетается с объектно-ориентированным подходом в программировании.

5. Разработан метод поддержки принятия решений при проектировании информационных систем предметной области ОТС, который отличается от существующих:

- ✓ применением на этапе обследования предметной области мультиагентного имитационного моделирования для анализа процессов организационно-технических систем;
- ✓ рассмотрением разрабатываемой распределенной информационной системы в виде мультиагентной системы;
- ✓ применением фреймово-семантической модели представления знаний на основе фрейм-концептов и концептуальных графов в целях формализации знаний о предметных областях разработки информационных систем и ОТС;
- ✓ интеграцией структурного и объектно-ориентированного подходов, мультиагентного подхода для решения задачи поддержки принятия решений при автоматизации процессов организационно-технических систем;
- ✓ преобразованием мультиагентной имитационной модели ОТС в основу модели архитектуры информационной системы и ее элементов, представлением архитектуры в виде структурных диаграмм и диаграмм объектно-ориентированного подхода для обеспечения эффективного взаимодействия между специалистами-предметниками и ИТ-специалистами;
- ✓ анализом эффективного распределения баз данных и знаний при наличии риска продолжительных отказов в обслуживании распределенной информационной системы.

6. На основе моделей и метода поддержки принятия решений в области создания информационных систем разработаны:

- ✓ интерфейсы системы поддержки принятия решений, ориентированные на конечного пользователя;
- ✓ программное, информационное, алгоритмическое и методическое обеспечение проблемно-ориентированного пакета BPsim.SD;
- ✓ технология работы с BPsim.SD.

7. Разработанная система поддержки принятия решений BPsim.SD отличается от существующих:

- ✓ интеграцией структурного и объектно-ориентированного подходов с имитационным моделированием мультиагентных процессов преобразования ресурсов;
- ✓ возможностью конвертировать структурные диаграммы в диаграммы объектно-ориентированного подхода;
- ✓ возможностью создания модели поведения программного агента;
- ✓ возможностью создания прототипа форм пользовательского интерфейса непрограммирующим пользователем.

Библиографический список

1. Colin J. Neill. Laplante Requirements Engineering: The State of the Practice / Colin J. Neill and Phillip A. // IEEE Software. 2003. November/December. P. 40–45.
2. Minsky M. A framework for Representing Knowledge in The Psychology of Computer Vision / M. Minsky; P. H. Winston (ed.). [S.l.]: McGraw-Hill, 1975. P.76.
3. Multi-agent approach for the metallurgical enterprise information system development / Aksyonov K. A. [et al.]//24th Int. Crimean Conference "Microwave & Telecommunication Technology" (CriMiCo'2014). 7–13 September, Sevastopol. Sevastopol, 2014. Vol. 1. P.437–438.
4. Sowa J. F. Knowledge Representation: Logical, Philosophical and Computational Foundations / Sowa J. F. Pacific Grove, CA: Brooks / Cole Publishing Co., 2000. 594 p.
5. Stephen A. White. BPMN Modeling and Reference Guide / Stephen A. White, Derek Miers. [S.l.] : Future Strategies Inc., 2008. 226 p.
6. Support for Analysis, Design and Implementation Stages with MASDK / Gorodetsky V. [et al.] // LNCS. 2009. 5386. P. 272–287.
7. UML — The Unified Modeling Language [Электронный ресурс] // Unified Modeling Language : [сайт]. URL: http://www.uml.org (дата обращения: 12.05.2016).
8. Wooldridge M. Intelligent Agents: Theory and Practice/Wooldridge M., Jennings N. // The Knowledge Engineering Review. 1995. Vol. 10, № 2. P. 115–152.
9. Аксенов К. А. Теория и практика средств поддержки принятия решений : монография / К. А. Аксенов. Saarbrucken (Germany): LAP LAMBERT Academic Publishing GmbH & Co. KG, 2011. 341 с.
10. Александров Д. В. Консалтинг при информатизации организаций : учеб. пособие / Д. В. Александров, Д. Н. Фадин. Владимир : Изд-во Владим. гос. ун-та, 2006. 72 с.

11. Бек Кент. Экстремальное программирование / Бек Кент. СПб. : Питер, 2002. 224 с.
12. Вендров А. М. Проектирование программного обеспечения экономических информационных систем / А. М. Вендров. М. : Финансы и статистика, 2005. 544 с.
13. Верников Геннадий. Стандарт онтологического исследования IDEF5 [Электронный ресурс] / Г. Верников.//Верников.ру: [сайт]. URL: http://vernikov.ru/krisis/item/31— idef5.html (дата обращения: 15.05.2016).
14. Инструментальные средства для разработки мультиагентных систем промышленного масштаба [Электронный ресурс] / Скобелев П. О. [и др.] // СамНЦ РАН : [сайт]. URL: http://www.ssc.smr.ru/media/ipuss_conf/06/5_01.pdf (дата обращения: 15.06.2016).
15. Разработка автоматизированной системы анализа, моделирования и принятия решений для металлургического предприятия на основе мультиагентного подхода / Аксенов К. А. [и др.] // Автоматизация в промышленности. 2014. № 7. С. 49–53.
16. Разработка методов, моделей и алгоритмов проектирования информационных систем для предметной области процессов преобразования ресурсов: отчет по проекту № 01.2.007 08048-/ООО «НПП «Системы автоматизации поддержки бизнеса»; Рук. К. А. Аксенов; П. П. Березовский, 2007. 55 с. Исполн.: И. А. Спицина, Е. Ф. Смолий [и др.].
17. Скобелев О. П. Открытые мультиагентные системы для оперативной обработки информации в процессах принятия решений : автореф. дис. ... д-р техн. наук / Скобелев О. П. Самара, 2003. 35 с.
18. Смирнова Г. Н. Проектирование экономических информационных систем / Г. Н. Смирнова, А. А. Сорокин, Ю. Ф. Тельнов. М. : Финансы и статистика, 2001. 510 с.
19. Спицина И. А. Метод поддержки принятия решений при разработке информационных систем на основе мультиагентного подхода : дис. ... канд. техн. наук: 05.13.10 / И. А. Спицина. Новосибирск : ФГОБУ ВПО СибГУТИ, 2015. 166 с.
20. Спицина И. А. Численный анализ задержек в синхронной информационной системе [Электронный ресурс] / И. А. Спицина, А. Л. Крохин // Междунар. научн.-техн. интернет-конф. «Информационные системы и технологии» (2013 год). URL: http://isit-conf.gu-unpk.ru/conferences/2/materials/index/415 (дата обращения: 10.06.2015).

21. Технология разработки прикладных многоагентных систем в инструментальной среде MASDK [Электронный ресурс] / Городецкий В. И. [и др.] // СПИИРАН. 2006. Т. 1, вып. 3. URL: http://www.proceedings.spiiras.nw.ru (дата обращения: 15.06.2016).
22. Хаммер М. Реинжиниринг корпорации (манифест революции в бизнесе) / М. Хаммер, Дж. Чампи. СПб. : С.-Петербург. ун-т, 1997. 332 с.
23. Швецов А. Н. Модели и методы построения корпоративных интеллектуальных систем поддержки принятия решений : дис. ... д-ра техн. наук: 05.13.01 / А. Н. Швецов. СПб., 2004. 461 с.


ИЗДАТЕЛЬСТВО УНИВЕРСИТЕТСКОГО КНИГОИЗДАНИЯ
УНИВЕРСИТЕТСКАЯ КНИГА
2015


9 785799 620387